BAB 2
TINJAUAN PUSTAKA

2.1 Sekilas Tempat Penelitian

AIRPLANE SYSTM adalah salah satu *pioneer distro* di Bandung yang sudah berdiri sejak awal-awal *distro* sedang menjamur di tahun 1998. Perusahaan yang bergerak di dunia industri kreatif ini mengambil nama Airplane karena sebagai simbol tercepat layanan yang paling modern dan canggih untuk semua orang yang diangkut ke daerah sasaran di era modern. Dengan layanan dan material yang kuat seperti halnya dalam pembuatan pesawat terbang, perusahaan ini berusaha keras membangun citra yang baik dan selalu *go public* bagi mereka yang dinamis dan hidup di era modern ini. Desain produk merupakan komitmen untuk kerja keras dan rasa hormat terhadap detail yang rumit, kuat, dan modis, dengan desain kualitas tinggi dan bahan. Semua ini disajikan dan terintegrasi dalam keinginan untuk memberikan solusi bagi kebutuhan gaya hidup remaja.

2.1.1 Logo Perusahaan

Logo dari AIRPLANE SYSTM dapat dilihat pada gambar 2.1 dibawah ini:



Gambar 2.1 Logo Perusahan AIRPLANE SYSTM

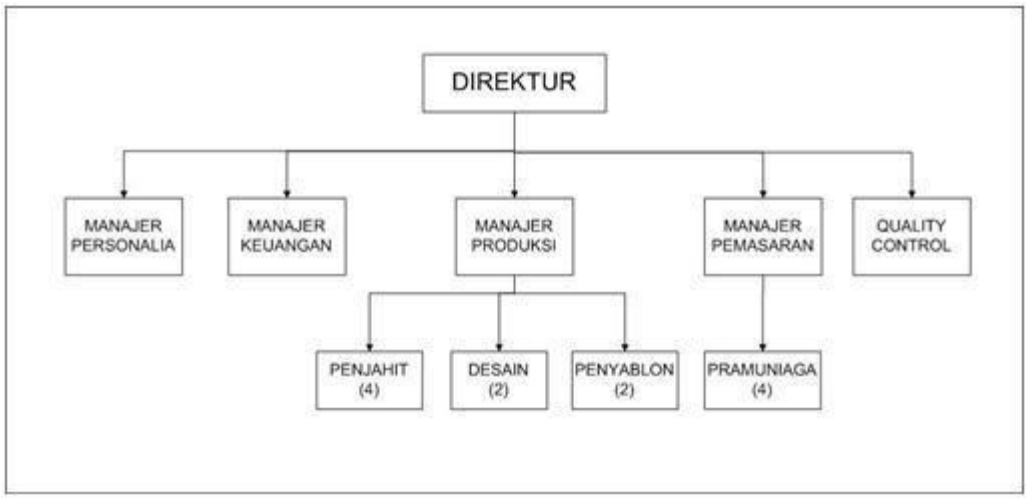
2.1.2 Visi dan Misi

Visi AIRPLANE SYSTM adalah memberikan warna baru dalam industri kreatif dan menjadi perusahaan desain produk distro terkemuka di Indonesia. Semua ini disajikan dan terintegrasi dalam keinginan untuk memberikan solusi bagi kebutuhan gaya hidup remaja.

AIRPLANE SYSTM mempunyai misi untuk memberikan karya-karya terbaik dengan sentuhan ide-ide kreatif agar dapat memenuhi kebutuhan di bidang *fashion* yang semakin lama semakin berkembang.

2.1.3 Struktur Organisasi Perusahaan

Berikut adalah struktur organisasi dari perusahaan AIRPLANE SYSTM, dapat dilihat pada gambar 2.2 dibawah ini :



Gambar 2.2 Struktur Organisasi Perusahaan

2.1.4 Deskripsi Kerja

Berikut ini adalah deskripsi kerja pada struktur organisasi yang nantinya akan terlibat didalam sistem. Hanya terdapat satu bagian yang terlibat dalam sistem yang akan dibangun, yaitu Manajer Pemasaran. Berikut adalah tugas dari Manajemen Pemasaran, yaitu :

1. Mempersiapkan rencana/strategi umum bagi perusahaan
2. Melaksanakan rencana tersebut

3. Mengadakan evaluasi, menganalisa dan mengawasi rencana tersebut dalam pelaksanaannya. (untuk mengukur hasil dan penyimpangannya serta untuk mengendalikan aktivitas).

2.2 Landasan Teori

Bagian ini berisi tentang beberapa teori-teori yang mendukung dalam proses analisis dan implementasi berdasarkan masalah yang diangkat dalam pembangunan sistem.

2.2.1 Pengertian Data, Informasi dan *Knowledge*

Data merupakan fakta yang dikumpulkan, disimpan, dan diproses oleh sebuah sistem informasi. Saat ini, akumulasi pertumbuhan jumlah data berjalan dengan cepat dalam format dan basis data yang berbeda. Data-data tersebut, antara lain [3]:

1. Data operasional atau transaksional, seperti penjualan, inventaris, penggajian, akuntansi, dan sebagainya.
2. Data nonoperasional, seperti industri penjualan (supermarket), peramalan, dan data ekonomi makro.
3. Metadata, adalah data mengenai data itu sendiri, seperti desain logika basis data atau definisi kamus data.

Sementara informasi adalah pola, asosiasi, atau hubungan antara semua data yang dapat memberikan informasi. Sebagai contoh, analisis titik eceran (*retail point*) data transaksi penjualan dapat menghasilkan informasi mengenai produk apa yang sebaiknya dijual dan kapan menjualnya.

Pengetahuan (*knowledge*) sebenarnya merupakan sebuah informasi juga yang merupakan hasil dari pengolahan data. Vercellis (2009) memandang bahwa suatu informasi dikatakan pengetahuan jika dapat digunakan dalam pengambilan keputusan. Dengan demikian pengetahuan dapat dijelaskan kembali sebagai kumpulan dari data dan informasi yang bertemu dengan kompetensi dan pengalaman seseorang untuk menindak lanjuti data dan informasi yang ada sehingga dapat dikembangkan untuk pengambilan suatu keputusan. Tidak seperti informasi yang hanya bersifat memberi tahu, pengetahuan harus mampu digunakan untuk proses pengambilan keputusan.

2.2.2 *Data Mining*

Berikut akan dijelaskan beberapa hal yang bersangkutan dengan *data mining*.

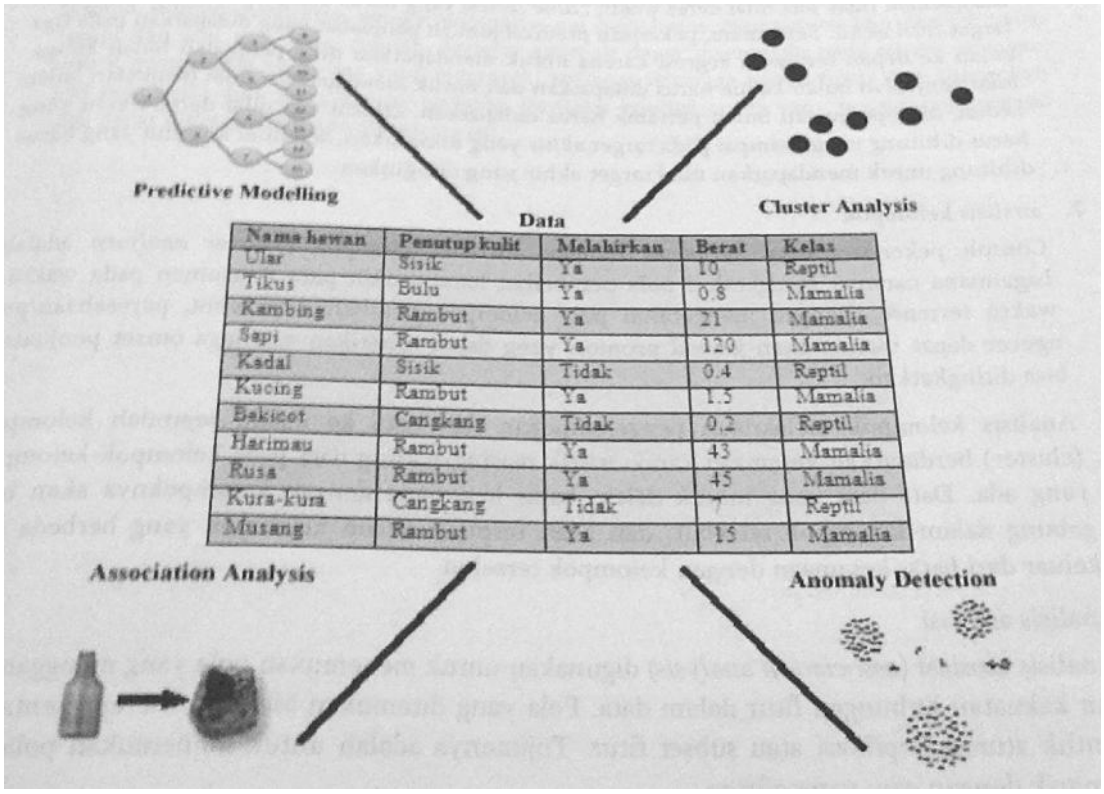
2.2.2.1 Pengertian *Data Mining*

Tan (2006) mendefinisikan *data mining* sebagai proses untuk mendapatkan informasi yang berguna dari gudang basis data yang besar. *Data mining* juga dapat diartikan sebagai pengekstrakan informasi baru yang diambil dari bongkahan data besar yang membantu dalam pengambilan keputusan.

Salah satu teknik yang dibuat dalam *data mining* adalah bagaimana menelusuri data yang ada untuk membangun sebuah model, kemudian menggunakan model tersebut agar dapat mengenali pola data yang lain yang tidak berada dalam basis data yang tersimpan. Kebutuhan untuk prediksi juga dapat memanfaatkan teknik ini. Dalam *data mining* pengelompokan data juga bisa dilakukan. Tujuannya adalah agar kita dapat mengetahui pola universal data-data yang ada. Anomaly data transaksi juga perlu dideteksi untuk dapat mengetahui tindak lanjut berikutnya yang dapat diambil. Semua hal tersebut bertujuan mendukung kegiatan operasional perusahaan sehingga tujuan akhir perusahaan diharapkan dapat tercapai.

2.2.2.2 Pekerjaan Dalam *Data Mining*

Pekerjaan yang berkaitan dengan *data mining* dapat dibagi menjadi empat kelompok, yaitu model prediksi (*prediction modelling*), analisis kelompok (*cluster analysis*), analisis asosiasi (*association analysis*), dan deteksi anomali (*anomaly detection*).



Nama hewan	Penutup kulit	Melahirkan	Berat	Kelas
Ular	Sisik	Ya	10	Reptil
Tikus	Bulu	Ya	0.8	Mamalia
Kambing	Rambut	Ya	21	Mamalia
Sapi	Rambut	Ya	120	Mamalia
Kadal	Sisik	Tidak	0.4	Reptil
Kucing	Rambut	Ya	1.5	Mamalia
Bekicot	Cangkang	Tidak	0.2	Reptil
Harimau	Rambut	Ya	43	Mamalia
Rusa	Rambut	Ya	45	Mamalia
Kura-kura	Cangkang	Tidak	7	Reptil
Musang	Rambut	Ya	15	Mamalia

Gambar 2.3 Pekerjaan Utama Data Mining

1. Model prediksi

   Model prediksi berkaitan dengan pembuatan sebuah model yang dapat melakukan pemetaan dari setiap himpunan variabel ke setiap targetnya, kemudian menggunakan model tersebut untuk memberikan nilai target pada himpunan baru yang didapat. Ada dua jenis model, yaitu klasifikasi dan regresi. Klasifikasi digunakan untuk variabel target diskret, sedangkan regresi untuk variabel target kontinu.

   Misalnya, pekerjaan untuk melakukan deteksi jenis penyakit pasien berdasarkan sejumlah nilai parameter penyakit yang diderita masuk dalam jenis klasifikasi karena di sini target yang diharapkan adalah diskret, hanya beberapa jenis kemungkinan nilai target yang didapatkan, tidak ada nilai deret waktu (*time series*) yang harus didapatkan untuk mendapatkan target nilai akhir. Sementara, pekerjaan prediksi jumlah penjualan yang didapatkan pada tiga bulan ke depan termasuk regresi karena untuk mendapatkan nilai penjualaan bulan

ketiga, nilai penjualaan bulan kedua harus didapatkan dan untuk mendapatkan nilai penjualan bulan kedua, nilai penjualan bulan pertama harus didapatkan. Di sini ada nilai deret waktu yang harus dihitung untuk sampai pada target akhir yang diinginkan, ada nilai kontinu yang harus dihitung untuk mendapatkan nilai target akhir yang diingingkan.

2. Analisis Kelompok

   Contoh pekerjaan yang berkaitan dengan analisis kelompok (*cluster analysis*) adalah bagaimana caranya mengetahui pola pembelian barang oleh para konsumen pada waktu-waktu tertentu. Dengan mengetahui pola kelompok pembelian tersebut, perusahaan/pengecer dapat menentukan jadwal promosi yang dapat diberikan sehingga omset penjualan bisa ditingkatkan.

   Analisis kelompok melakukan pengelompokan data-data ke dalam sejumlah kelompok (*cluster*) berdasarkan kesamaan karakteristik masing-masing data pada kelompok-kelompok yang ada. Data-data yang masuk dalam batas kesamaan dengan kelompoknya akan bergabung dalam kelompok tersebut, dan akan terpisah dalam kelompok yang berbeda jika keluar dari batas kesamaan dengan kelompok tersebut.

3. Analisis asosiasi

   Analisis asosiasi (*association analysis*) digunakan untuk menentukan pola yang menggambarkan kekuatan hubungan fitur dalam data. Pola yang ditemukan biasanya merepresentasikan bentuk aturan implikasi atau subset fitur. Tujuannya adalah untuk menemukan pola yang menarik dengan cara yang efisien.

   Penerapan yang paling dekat dengan kehidupan sehari-hari adalah analisis data keranjang belanja. Sebagai contoh, pembeli adalah ibu rumah tangga yang akan membeli kebutuhan rumah tangga di sebuah supermarket. Jika ibu tersebut membeli beras, sangat besar kemungkinannya bahwa ibu itu juga membeli barang lain, misalnya

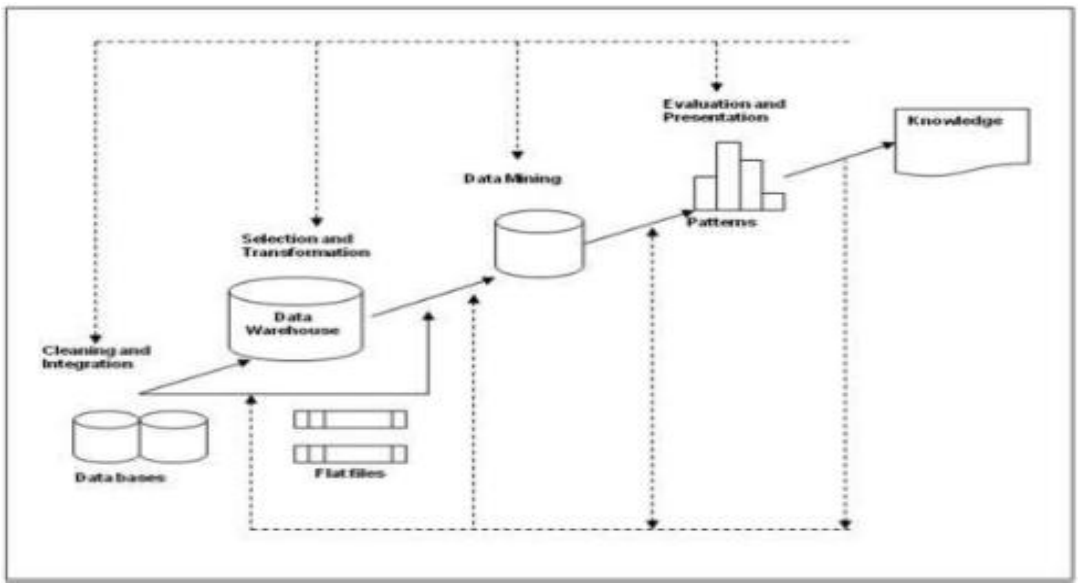
minyak, telur, dan tidak mungkin atau jarang membeli barang lain seperti topi atau buku. Dengan mengetahui hubungan yang lebih kuat antara beras dengan telur daripada beras dengan topi, pengecer dapat menentukan barang-barang yang sebaiknya disediakan dalam jumlah yang cukup banyak.

4. Deteksi Anomali

   Pekerjaan deteksi anomali (*anomaly detection*) berkaitan dengan pengamatan sebuah data dari sejumlah data yang secara signifikan mempunyai karakteristik yang berbeda dari sisa data yang lain. Data-data yang karakteristiknya menyimpang (berbeda) dari data yang lain disebut *outlier.* Algoritma deteksi anomali yang baik harus mempunyai laju deteksi yang tinggi dan laju eror yang rendah. Deteksi anomali dapat diterapkan pada sistem jaringan untuk mengetahui pola data yang memasuki jaringan sehingga penyusupan bisa ditemukan, jika pola kerja data yang datang berbeda. Perilaku kondisi cuaca yang mengalami anomali juga dapat dideteksi dengan algoritma ini.

2.2.2.3 Konsep *Data Mining*

Data mining sangat diperlukan terutama dalam mengelola data yang sangat besar untuk memudahkan aktifitas *recording* suatu transaksi dan untuk proses *data warehousing* agar dapat memberikan informasi yang akurat bagi pengguna *data mining.* Alasan utama *data mining* sangat dibutuhkan dalam industri informasi karena tersedianya data dalam jumlah yang besar dan semakin besarnya kebutuhan untuk mengubah data tersebut menjadi informasi dan pengetahuan yang berguna untuk mengubah data tersebut menjadi informasi dan pengetahuan yang berguna karena sesuai fokus bidang ilmu ini yaitu melakukan kegiatan mengekstraksi atau menambang pengetahuan dari data uang berukuran aatau berjumlah besar. Informasi inilah yang nantinya sangat berguna untuk pengembangan. Berikut adalah langkah-langkah dalam data mining [4]:



Gambar 2.4 *Konsep* Data Mining

Berikut adalah tahap-tahap dalam *Data Mining* :

1. *Data cleaning* yaitu untuk menghilangkan *noise data yang tidak konsisten.*
2. *Data integration* yaitu menggabungkan beberapa *file* atau *database.*
3. *Data selection* yaitu data yang relevan dengan tugas analisis dikembalikan ke dalam database untuk proses *data mining.*
4. *Data transformation* yaitu data berubah atau bersatu menjadi bentuk yang tepat untuk menambang dengan ringkasan performa atau operasi agresi.
5. *Data mining* yaitu proses esensial dimana metode yang intelejen digunakan untuk mengekstrak pola data.
6. *Knowledge Discovery* yaitu proses esensial dimana metode intelejen digunakan untuk mengekstrak pola data.
7. *Pattern evolution* yaitu untuk mengidentifikasi pola yang benar-benar menarik yang mewakili pengetahuan berdasarkan atas beberapa tindakan yang menarik.
8. *Knowledge presentation* yaitu gambaran teknik visualisasi dan pengetahuan digunakan untuk memberikan pengetahuan yang telah ditambah kepada *user*.

2.2.2.4 Tahapan *Data Mining*

Tahapan dalam melakukan *data mining* salah satunya adalah *preprocessing* data. Tahapan ini biasanya diperlukan karena data yang akan digunakan belum baik, yang disebabkan oleh beberapa faktor berikut ini [4]:

1. *Incomplete* : tidak lengkapnya nilai suatu atribut, tidak lengkapnya atribut-atribut yang penting, atau hanya mempunyai data yang merupakan rekapitulasi.
   Contoh : pekerjaan=” ‘’
   Hal tersebut dapat disebabkan oleh perbedaan kebijakan ketika dapat dianalisa, bias juga disebabkan oleh permasalahan yang ditimbulkan oleh manusia, *hardware*, atau *software*.
2. *Noisy* : mengandung *error* atau merupakan *value* yang tidak wajar.
   Contoh : gaji ”-100’’
   Timbul karena kesalahan *entry* oleh manusia atau *computer error*, atau karena terdapat kesalahan ketika proses pengiriman data.
3. *Inconsisten **:*** mengandung nilai yang saling bertentangan.
   Contoh : umur =”42” dan ulang tahun = “02/10/1981”
   Masalah ini muncul karena perbedaan sumber data, karena pada *data mining* data didapatkan dari banyak sumber dan sangat mungkin terdapat perbedaan persepsi pengelolahan data. Seail itu, perbedaan ini muncul karena pelanggaran terhadap *fungsional dependency*, misalnya melakukan perubahan pada data yang terhubung dengan data lain.

Sebenarnya data yang bisa diterima untuk bisa diproses menjadi informasi atau *knowledge* adalah data yang mempunyai kualitas diantaranya :

1. Akurat
2. Lengkap
3. Konsisten
4. Relevan
5. Bisa dipercaya
6. Mempunyai nilai tambah

7. Kemudahan untuk dimengerti

Jika data tidak dalam kualitas seperti yang telah diuraikan diatas, maka kualitas analisis data menjadi kurang sehingga hasilnya pun kurang bermakna. Hal tersebut harus dihindarkan karena hasil analisa yang salah dapat berujung pada solusi yang salah. Untuk itu, perlu dilakukan preprocessing data yangbertujuan untuk agar membuat data menjadi lebih berkualitas. Adapun tahapan-tahapannya adalah sebagai berikut :

1. *Data Cleaning* : mengisi/mengganti nilai-nilai yang hilang, menghaluskan data yang *noisy*, mengidentifikasi dan menghilangkan data yang tidak wajar, dan menyelesaikan masalah *inconsistensi* data.
2. *Data Integration* : menggabungkan beberapa *database* dan *file* menjadi satu sehingga didapatkan sumber data yang besar.
3. *Data Transformation :* normalisasi dan agregasi data.
4. *Data Reduction :* mengurangi *volume* data namun tetap mempertahankan arti dalam hal hasil analisis data.
5. *Data Discretization :* merupakan bagian dari data *reduction* dengan memperhitungkan data yang signifikan, khususnya pada *data numeric*.

2.2.3 Association Rule

Aturan asosiasi *(Association rule)* adalah salah satu teknik tentang ‘apa bersama apa’. Ini bisa berupa transaksi di supermarket, misalnya seseorang yang membeli susu bayi juga membeli sabun mandi. Di sini berarti susu bayi bersama sabun mandi. Karena awalnya berasal dari studi tentang *database* transaksi pelanggan untuk menentukan kebiasaan suatu produk dibeli bersama produk apa, maka aturan asosiasi juga sering dinamakan *Market Basket* [5].

Untuk mencari *association rule* dari suatu kumpulan data, tahap pertama yang harus dilakukan adalah mencari *frequent itemset* terlebih dahulu. *Frequent itemset* adalah sekumpulan *item* yang sering muncul secara bersamaan. Setelah semua pola *frequent itemset* ditemukan, barulah mencari aturan asosiatif atau aturan keterkaitan yang memenuhui syarat yang telah ditentukan.

Jika diasumsikan bahwa barang yang dijual di swalayan adalah semesta, maka setiap barang akan memiliki *Boolean variable* yang akan menunjukan keberadaannya atau tidak barang tersebut dalam satu transaksi atau satu keranjang belanja. Pola *Boolean* yang didapat digunakan untuk menganalisa barang yang dibeli secara bersamaan. Pola tersebut dirumuskan dalam sebuah *association rule.* Sebagai contoh konsumen biasanya akan membeli kopi dan susu yang ditujukan sebagai berikut :

Kopi → susu [*support* = 2%, *confidence* = 60%]

Nilai *support* 2% menunjukan bahwa keseluruhan dari total transaksi konsumen membeli kopi dan susu secara bersamaan yaitu sebanyak 2%. Sedangkan *confidence* 60% yaitu menunjukan bila konsumen membeli kopi dan pasti membeli susu sebesar 60%.

Penting tidaknya suatu aturan asosiatif dapat diketahui dengan dua parameter, yaitu *support* dan *confidence. Support* (nilai penunjang) adalah presentase kombinasi *item* tersebut dalam *database*, sedangkan *confidence* (nilai kepastian) adalah kuatnya hubungan antar-*item* dalam aturan asosiasi.

Dalam menentukan suatu *association rule*, terdapat suatu ukuran kepercayaan yang didapat dari hasil pengelolahan data dengan perhitungan tertentu. Umumnya ada dua ukuran, yaitu:

a. *Support* : suatu ukuran yang menunjukan seberapa besar tingkat dominasi suatu *itemset* dari keseluruhan transaksi. Ukuran ini menentukan apakah suatu *itemset* layak untuk dicari *confidence*-nya (misal, dari keseluruhan transaksi yang ada, seberapa besar tingkat dominasi suatu *item* yang menunjukan bahwa *item* A dan *item* B dibeli bersamaan).
b. *Confidence* : suatu ukuran yang menunjukan hubungan antara 2 *item* secara *conditional* (misal, menghitung kemungkinan seberapa sering *item* B dibeli oleh pelanggan jika pelanggan tersebut membeli sebuah *item* A).

Kedua ukuran ini nantinya berguna dalam menentukan kekuatan suatu pola dengan membandingkan pola tersebut dengan nilai minimum kedua parameter tersebut yang ditentukan oleh pengguna. Analisis asosiasi didefinisikan suatu proses untuk menemukan semua aturan asosiasi yang memenuhi syarat

minimum untuk *support* (*minimum support*) dan syarat minimum untuk *confidence* (*minimum confidence*).

Nilai *support* sebuah *item* diperoleh dengan rumus sebagai berikut[6] :

$$Support(A) = \frac{Jumlah\ transaksi\ mengandung\ A}{Total\ Transaksi} x100\% \text{..... Persamaan (2.1)}$$

Sementara itu, nilai *support* dari 2 *item* diperoleh dari rumus berikut:

$$Support\ (A,B) = \frac{\text{Jumlah transaksi yang mengandung A dan B}}{\text{Jumlah transaksi}} x100\% \text{..Persamaan (2.2)}$$

Sedangkan nilai *confidence* dapat dicari setelah pola frekuensi munculnya sebuah *item* ditemukan. Rumus untuk menghitung *confidence* adalah sebagai berikut:

$$Confidence = \frac{\text{Jumlah transaksi yang mengandung A dan B}}{\text{Jumlah transaksi yang mengandung A}} x100\% \text{.....Persamaan(2.3)}$$

Berikut adalah contoh penerapan metode *association rule* :

Misalkan diberikan tabel data transaksi sebagai berikut dengan minimum *support count* = 2

Tabel 2.1 Data Transaksi

TID	*Items*
1	B
2	B, D
3	A, C, D, E
4	A, D, E
5	B, C
6	A, C
7	A

Yang mana nilai *support count* didapat dari : $\frac{15}{7}$ x 100% = 2.14 ≈ 2

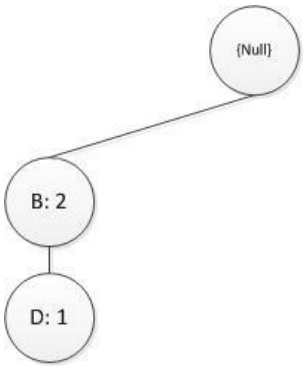
Frekuensi kemunculan tiap item dapat dilihat pada tabel berikut:

Tabel 2. 2 Frekuensi Kemunculan Tiap Karakter

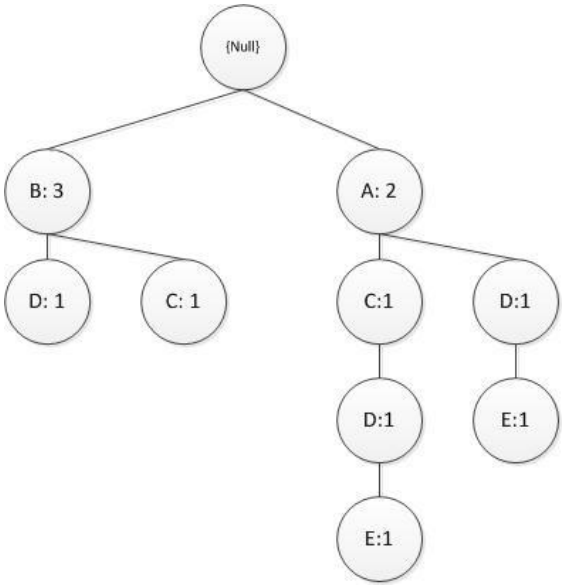
Item	Frekuensi
A	4

B	3
C	3
D	3
E	2

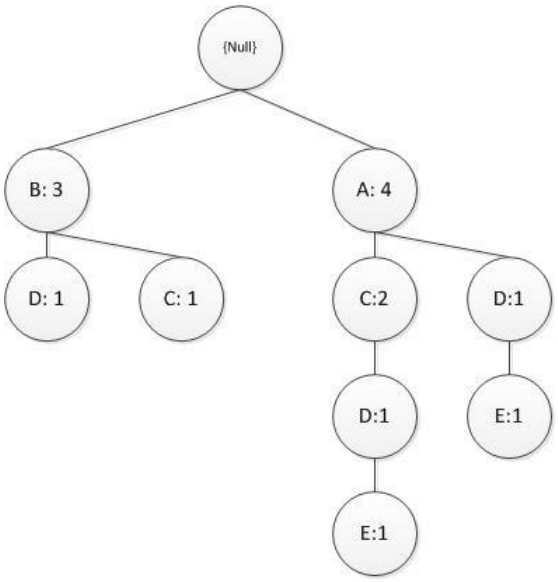
Karena contoh kasus dalam kemunculan *item* sudah *frequent* dalam setiap transaksi, sudah diurut berdasarkan frekuensi yang paling tinggi. Selanjutnya pada gambar di bawah ini memberikan ilustrasi mengenai pembentukan FP-Tree setelah pembacaan TID 2



Gambar 2.5 Hasil Pembentukan FP-Tree **setelah pembacaan TID 2**



Gambar 2.6 Hasil Pembentukan FP-Tree **setelah pembacaan TID 5**



Gambar 2.7 Hasil pembentukan FP-Tree **setelah pembacaan TID 7**

Diberikan 7 data transaksi dengan 5 jenis item seperti pada tabel di atas. Gambar 5-7 menunjukan proses terbentuknya FP-Tree setiap TID dibaca. Setiap simpul pada FP-Tree mengandung nama sebuah item dan *counter support* yang berfungsi untuk menghitung frekuensi kemunculan item tersebut dalam tiap lintasan transaksi.

Setelah FP-Tree terbentuk dari sekumpulan data transaksi maka, proses pencarian *frequent itemset* dengan menggunakan algoritma FP-Growth akan dilakukan yang dibagi menjadi tiga tahapan utama, yaitu:

a. Tahap pembangkitan *conditional pattern base*

   Merupakan subdata yang berisi *prefix path* (lintasan awal) dan *suffix pattern* (pola akhiran). Pembangkitan *conditional pattren base* didapatkan melalui FP-Tree yang telah dibangun sebelumnya.

b. Tahap pembangkitan *conditional* FP-Tree

   Pada tahap ini, *support count* dari setiap *item* pada setiap *conditional pattern base* dijumlahkan, lalu setiap *item* yang memiliki jumlah *support count* lebih besar atau sama dengan minimum *support count* akan dibangkitkan dengan *conditional* FP-Tree seperti pada tabel 2.3 dibawah ini.

Tabel 2.3 Hasil Perhitungan Conditional Pattern Base dan Conditional FP-Tree

Item	*Conditional Pattern Base*	*Conditional* FP Tree
E	{{A,C,D:1}, {A,D:1}}	{<A:2,D:2>}
D	{{B:1}, {A,C:1}, {A:1}}	{A:2}
C	{B:1}, {A:2}	{A:2}

c. Tahap pencarian *frequent itemset.*

   Apabila *conditional* FP-Tree merupakan lintasan tunggal(*single-path*), maka didapatkan *frequent itemset* dengan melakukan kombinasi *item* untuk setiap *conditional* FP-Tree. Jika bukan lintasan tunggal, maka dilakukan pembangkitan FP-Growth secara *rekursif* (proses memanggil dirinya sendiri) seperti contoh pada tabel 2.4 dibawah ini.

Tabel 2.4 *Hasil* Frequent Itemset

Suffix	*Frequent Itemset*
E	E, A-E, D-E, A-D-E
D	D, A-D
C	C, A-C

Tabel 2.5 *Hasil* Generate Rule

Rule	*Support*	*Confidence* (%)
A → E	2	(2/4)*100 = 50
E → A	2	(2/2)*100 = 100
A → D	2	(2/4)*100 = 50
D → A	2	(2/3)*100 = 66.67
A → C	2	(2/4)*100 = 50
C → A	2	(2/3)*100 = 66.67
D → E	2	(2/3)*100 = 66.67
E → D	2	(2/2)*100 = 100
A- D →E	2	(2/2)*100 = 100
D- E →A	2	(2/2)*100 = 100

2.2.4 Algoritma FP- Growth

Algoritma yang sama dengan Apriori, FP-Growth mulai dengan menghitung item tunggal sesuai dengan jumlah kemunculan item yang ada didalam *dataset*. Setelah proses penghitungan selesai maka akan dibuat struktur pohon pada tahap kedua. Pohon yang dibuat mulanya kosong yang nanti akan diisi dengan hasil dari dataset yang telah didapat sebelumnya. Kunci untuk mendapatkan struktur pohon yang bisa didapatkan dengan proses lebih cepat untuk mencari item set yang besar menjadi sedikit dengan di urutkan secara *descending* dari frekuensi yang ada *dataset* tersebut. Masing-masing item yang tidak mencapai kebutuhan minimum dari *threshold* tidak dimasukkan kedalam pohon, tapi dikeluarkan secara efektif dari *dataset* [8].

FP-Tree merupakan struktur penyimpanan data yang dimampatkan. FP-Tree dibangun dengan memetakan setiap data transaksi ke dalam setiap lintasan tertentu dalam FP-Tree. Karena dalam setiap transaksi yang dipetakan, mungkin ada transaksi yang memiliki item yang sama, maka lintasannya memungkinkan untuk saling menimpa. Semakin banyak data transaksi yang memiliki item yang sama, maka proses pemampatan dengan struktur data FP-Tree semakin efektif. Kelebihan dari FP-Tree adalah hanya memerlukan dua kali pemindaian data transaksi yang terbukti sangat efisien.

Adapun FP-Tree adalah sebuah pohon dengan definisi sebagai berikut:

a. FP-Tree dibentuk oleh sebuah akar yang diberi label *null*, sekumpulan berupa pohon yang beranggotakan item-item tertentu, dan sebuah tabel *frequent header*.
b. Setiap simpul dalam FP- Tree mengandung tiga informasi penting, yaitu label item, menginformasikan jenis item yang direpresentasikan simpul tersebut, *support count*, merepresentasikan jumlah lintasan transaksi yang melalui simpul tesebut, dan *pointer* penghubung yang menghubungkan simpul-simpul dengan label item sama antar-lintasan, ditandai dengan garis panah putus-putus.

2.2.4.1 Langkah-Langkah Proses Perhitungan Association Rule Dengan Algoritma FP-Growth

Proses perhitungan *association rule* terdiri dari beberapa tahap adalah sebagai berikut [9] :

1. Membuat *Header Item*

   Header dalam hal ini selain sebagai *header* suatu *item* ke FP-Tree juga sebagai jenis *item* dasar yang memenuhi minimum *support*. Setelah mendapatkan *item* dan nilai *support*-nya, maka *item* yang tidak *frequent* dibuang dan *item* diurutkan berdasarkan nilai *support*-nya. Header

untuk item, disiapkan pada suatu *array* tertentu dan ditambahkan ketika membuat FP-Tree.

2. Membuat FP-Tree

   FP-Tree dibangun dengan mencari *item* sesuai urutan pada *item* yang *frequent*. Data transaksi tidak perlu diurutkan, dan untuk tiap *item* yang ditemukan bisa langsung dimasukkan ke dalam FP-Tree. Sesudah membuat *root*, tiap *item* yang ditemukan dimasukkan berdasarkan *path* pada FP-Tree. Jika *item* yang ditemukan sudah ada, maka nilai *support item* tersebut yang ditambahkan. Namun jika *path* belum ada, maka dibuat *node* baru untuk melengkapi *path* baru pada FP-Tree tersebut. Hal ini dilakukan selama item pada transaksi masih ada yang *qualified*, artinya memenuhi nilai minimum *support*. Jadi, *item-item* yang ditemukan dalam transaksi akan berurutan memanjang ke bawah. Dalam struktur FP-Tree, diterapkan alur *path* dari *child* hingga ke *root*. Jadi, suatu *path* utuh dalam FP-Tree adalah dari *child* terbawah hingga ke *root*. Tiap *node* pada FP-Tree memiliki *pointer* ke *parent*, sehingga pencarian harus dimulai dari bawah.

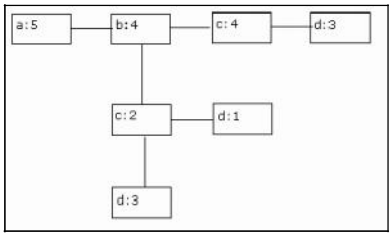
3. *Pattern Extraction*

   Pattern extraction dilakukan berdasarkan keterlibatan *item* pada suatu *path*. Di setiap *path*, diperiksa semua kombinasi yang mungkin dimana *item* tersebut terlibat. Di iterasi berikutnya dilakukan dengan melibatkan *item* berikutnya, tanpa melibatkan *item* sebelumnya, sehingga *pattern* yang sama tidak akan ditemukan dua kali pada *path* yang sama. Bila *item* pertama suatu hasil kombinasi bukan *item* terakhir (sebelum *root*), maka kombinasi *itemset* tersebut masih bisa dikembangkan lagi.

4. Memasukkan setiap pattern yang ditemukan ke dalam *PatternTree*

   Setelah mengolah FP-Tree menjadi *pattern-pattern*, diperlukan proses akumulasi *pattern-pattern* yang ditemukan mengingat *pattern* yang sama dapat ditemukan pada path yang berbeda. Untuk itu digunakan struktur data *Pattern Tree* (lihat Gambar 2.5). Setiap node

di *Pattern Tree* merepresentasikan dan menyimpan frekuensi suatu *pattern*. *Pattern Tree* terdiri atas *Pattern TreeNode* yang menyimpan nilai item, nilai *support* dan dilengkapi dengan dua *pointer* yaitu untuk horisontal dan vertikal.



Gambar 2.8 *Pattern Tree*

Misalnya pada *node* d:1 di atas, berarti terdapat *pattern* a-c-d bernilai *support* 1. Kemudian bila ada pattern a-c-d lagi bernilai *support* n yang ditemukan dari FP-Tree maka nilai *support* 1 tersebut menjadi n+1. Contoh hasil lengkap dari *PatternTree* tersebut:

1. a:5 menggambarkan bahwa ada pattern a sebanyak 5
2. b:4 menggambarkan bahwa ada pattern a-b sebanyak 4
3. c:4 menggambarkan bahwa ada pattern a-b-c sebanyak 4
4. d:3 menggambarkan bahwa ada pattern a-b-c-d sebanyak 3
5. c:2 menggambarkan bahwa ada pattern a-c sebanyak 2
6. d:1 menggambarkan bahwa ada pattern a-c-d sebanyak 1
7. d:3 menggambarkan bahwa ada pattern a-d sebanyak 3

5. Mengurutkan dan Menyeleksi *Pattern*

   Pattern yang tidak memenuhi minimum *support*, dihapus dari daftar *pattern*. *Pattern-pattern* yang tersisa kemudian diurutkan untuk memudahkan pembuatan *rules*.

2.2.5 Alat-Alat Pemodelan Sistem

Berikut ini adalah penjelasan dari alat-alat yang digunakan dalam pemodelan sistem yang akan dibangun.

2.2.5.1 *Entity Relation Diagram* (ERD)

ERD adalah suatu model jaringan yang menggunakan susunan data yang disimpan dalam sistem secara abstrak. Basis data relasional adalah kumpulan dari relasi-relasi yang mengandung seluruh informasi berkenaan suatu entitas/objek yang akan disimpan di dalam database. Tiap relasi disimpan sebagai sebuah *file* tersendiri. Perancangan basis data merupakan suatu kegiatan yang setidaknya bertujuansebagai berikut :

1. Menghilangkan redudansi data.
2. Meminimumkan jumlah relasi di dalam basis data.
3. Membuat relasi berada
4. Dalam bentuk normal, sehingga dapat meminimumkan permasalahan berkenaan dengan penambahan, pembaharuan dan penghapusan.

2.2.5.2 Diagram Konteks

Diagram konteks adalah diagram yang terdiri dari suatu proses dan menggambarkan ruang lingkup suatu sistem. Diagram konteks merupakan level tertinggi dari *Data Flow Diagram* yang menggambarkan seluruh *input* ke sistem atau *output* dari sistem. Diagram konteks memberi gambaran tentang keseluruan sistem. Sistem dibatasi oleh *boundary* (dapat digambarkan dengan garis putus). Dalam diagram konteks hanya ada satu proses [10].

2.2.5.3 Data Flow Diagram

Data flow diagram atau DFD adalah alat untuk menggambarkan komponen-komponen sebuah sistem, aliran data pada sistem, dimana data berasal, tujuan data serta penyimpanan data. Ada 4 komponen utama dalam DFD yaitu [9]:

1. Entitas eksternal (*External Entity*)
2. Arus data (*Data Flow*)
3. Proses (*Process*)
4. Penyimpanan data (*Data Store*)

DFD dapat dikatakan turunan dari diagram konteks, analisis lebih detail dari sistem dapat dilakukan dengan menurunkan DFD ke level 1,2,3 dan

seterusnya. Penjelasan simbol pada diagram konteks dan DFD dapat dilihat di daftar simbol.

2.2.5.4 Spesifikasi Proses

Spesifikasi proses digunakan untuk mendeskripsikan proses yang terjadi pada level yang paling dasar dalam DFD. Spesifikasi proses berfungsi mendeskripsikan tahapan yang dilakukan untuk merubah masukan menjadi keluaran. Spesifikasi proses ditampilkan dalam bentuk *form*, logika proses yang ditulis pada spesifikasi proses harus jelas [10]. Berikut adalah poin yang dideskripsikan pada spesifikasi proses :

1. No Proses : Menyatakan nomor proses
2. Nama Proses : Menyatakan nama proses
3. Sumber : Menyatakan sumber data yang menuju ke proses
4. *Input* : Menyatakan isi data yang masuk ke proses
5. *Output* : Menyatakan informasi yang keluar dari proses
6. Tujuan : Menyatakan tujuan informasi keluaran dari proses
7. Logika Proses : Menyatakan algoritma dari proses

2.2.5.5 Kamus Data

Kamus data adalah deskripsi formal mengenai seluruh komponen-komponen yang ada pada data flow diagram. Pada tahap perancangan, komponen-komponen pada kamus data akan menjadi bahan untuk menyusun basis data[10]. Berikut adalah poin yang dideskripsikan pada kamus data :

1. Nama aliran data : Data yang digunakan
2. *How used* : Proses apa saja yang digunakan
3. Keterangan : Uraian singkat data yang digunakan
4. Struktur data : Daftar komponen yang ada pada data yang digunakan
5. Deskripsi : Jenis data dalam representasi komputer untuk masing-masing data

2.2.6 Microsoft Visual Studio

Visual studio .Net adalah sekumpulan alat pengembangan *software* yang diperuntukkan bagi .Net platform[11]. Microsoft 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. 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).

C# sering dianggap sebagai bahasa penerus C++ atau versi canggih dari C++. C# adalah sebuah bahasa pemrograman yang dikembangkan oleh Microsoft dan menjadi salah satu bahasa pemrograman yang mendukung .Net programming melalui Visual Studio. C# didasarkan pada bahasa pemrograman C++, C# juga memiliki kemiripan dengan beberapa bahasa pemrograman seperti Visual Basic, Java, Delphi dan tentu saja C++.

2.2.7 MySQL

MySQL adalah *multiuser database* yang menggunakan bahasa *structured Query Language* (SQL). MySQL dalam operasi *client server* melibatkan *server daemon* MySQL di sisi *server* dan berbagai macam program serta *library* yang berjalan di sisi *client.* MySQL mampu menangani data yang cukup besar. Perusahaan yang mengembangkan MySQL yaitu TcX, mengaku bahwa MySQL yang mampu menyimpan data lebih dari 40 *database*, 10.000 tabel dan sekitar 7 juta baris, totalnya kurang lebih 100 *Gigabyte* data [12].

SQL adalah bahasa standar yang digunakan untuk mengakses *database server*. Bahasa ini pada awalnya dikembangkan oleh IBM, namun telah diadopsi dan digunakan sebagai bahasa yang standar industri. Dengan menggunakan SQL, proses akses *database* menjadi lebih *user-friendly* dibandingkan dengan menggunakan DBASE atau *Clipper* yang masih menggunakan perintah-perintah pemrograman.