BAB 2
TINJAUAN PUSTAKA

2.1 Profil Perusahaan

Mini market barokah yang berlokasi di jalan raya darangdan purwakarta merupakan salah satu minimarket yang menjual berbagai macam barang kebutuhan sehari-hari. Selain menjual barang-barang untuk kebutuhan sehari-hari dalam bentuk satuan, minimarket barokah juga menjual barang untuk kebutuhan sehari-hari dalam bentuk paket barang dengan harga jual lebih murah dibandingkan dengan membeli barang dalam bentuk satuan.

2.1.1 Sejarah Minimarket Barokah

Awal mula terbentuknya minimarket barokah berawal dari konsep/bentuk tradisional market atau dikenaal dengan toko kelontongan dengan nama toko barokah. Dengan berjalannya waktu toko ini mengalami perubahan bentuk/ konsep menjadi *modern* market atau lebih dikenal dengan mini market barokah. Perubahan konsep ini didasarkan pada perkembangan pada dunia retail dan tuntutan dari *stakholder*.

Pada tanggal 15 maret 2005, toko barokah resmi berubah konsep menjadi *modern markets* dengan nama mini market barokah perubahan ini mencakup penerapan aplikasi sistem ritel yang terintegrasi , rak *display standard* mini market, struktur organisasi modern market dan pengadaan perlengkapan mini market seperti komputer dan printer POS, *sccaner barkod*, *printer barcode* label dan sebagainya.

2.1.2 Logo

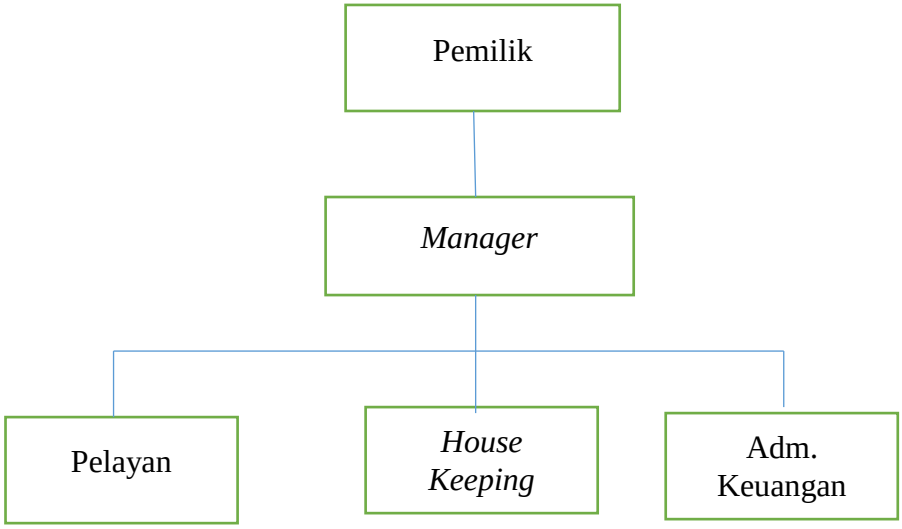
Logo dari toko barokah adalah terdapat nama barokah yang artinya berkah (selalu mengharapkan rezeki yang berkah dari Allah SWT),warna merah artinya keberanian dalam medobrak pasar artinya beda dalam segi pelayanan,harga dan kualitas.warna kuning melambangkan keceriaan dan ketulusan artinya selalu tulus dan setia dalam melayani dapat dilihat pada gambar 2.1.



Gambar 2. 1 Logo Mini Market Barokah

2.1.3 Struktur Organisasi

Struktur organisasi adalah gambaran yang menerangkan posisi dan hierarki kerja di dalam suatu perusahaan. Struktur organisasi Mini market Barokah dapat dilihat pada Gambar 2.1.3:



Gambar 2. 2 Struktur Organisasi Mini Market Barokah

Deskripsi tugas:

1. Pemilik bertanggung jawab penuh perusahaan , tugas dan tanggung jawabnya sebagai berikut:
   a. Membuat perencanaan, strategi dan kebijakan yang menyangkut operasi perusahaan.
   b. Menyusun anggaran kebutuhan persediaan barang.
   c. Melakukan kontrol secara keseluruhan atau operasi perusahaan.
   d. Memegang kendali atas keputusan penting yang bersifat umum berkaitan dengan keuangan.
2. Manager berfungsi untuk membantu pemilik, tugas dan tanggung jawabnya adalah sebagai berikut:
   a. Membantu pemilik dalam mengawasi perusahaan.
   b. Mengatur setiap bagian yang ada di perusahaan agar menjalankan tugasnya dengan baik.
3. Bagian pelayan
   a. Membantu dalam melayani pesanan-pesanan pelanggan.
   b. Melayani semua pesanan pelanggan.
4. Bagian *house keeping*
   a. Bertanggung jawab terhadap kebersihan, baik di luar ataupun di dalam lingkungan mini market.
   b. Membereskan dan membersihkan peralatan yang telah digunakan
5. Bagian adm.keuangan
   a. Bertanggung jawab terhadap hal-hal yang menyangkut keuangan.
   b. Menghitung pemasukan dan pengeluaran setiap bulannya.
   c. Melayani pelanggan dalam pembayaran transaksi.

2.1.4 Visi dan Misi

Visi dari mini market barokah adalah mendirikan minimarket dengan konsep modern market lokal yang berkedudukan dan berkepanjangan. Sedangkan Misi dari mini market Barokah adalah sebagai berikut:

1. Mengutamakan pelayanan terbaik untuk pelanggan serta kualitas makanan yang akan dijual.
2. Melakukan inovasi terhadap barang dan sistem penjualan secara berkala.
3. Mengembangkan usaha mini market hingga dapat membuka cabang ditempat lain.
4. Meningkatkan nama barangtifitas mini market dari sumber daya manusianya agar mini market terus berkembang ke arah yang lebih baik.

2.2 Landasan Teori

Landasan teori membahas mengenai materi atau teori apa saja yang digunakan sebagai acuan dalam membuat tugas akhir ini. Landasan teori yang di uraikan merupakan hasil studi literature, baik dari buku, maupun situs internet.

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, symbol teks, gambar, bunyi, atau kombinasinya [3]. Dalam pendekatan basis data tidak hanya berisi basis data itu sendiri tetapi juga termasuk definisi atau deskripsi dari data yang disimpan. Definisi data disimpan dalam sistem katalog, yang berisi informasi tentang struktur tiap berkas, tipe dan format penyimpanan tiap item data, dan berbagai konstrin dari data. Semua informasi yang disimpan dalam katalog ini biasa disebut meta-data [4].

2.2.2 Basis Data

Basis Data terdiri atas 2 kata, yaitu Basis dan Data. Basis kurang lebih dapat diartikan sebagai markas atau gudang, tempat bersarang/berkumpul. Sedangkan Data adalah representasi fakta dunia nyata mewakili suau objek seperti manusia pegawai, siswa, pembeli, pelanggan), barang, hewan, peristiwa, konsep,keadaan, dan sebagainya, yang diwujudkan dalam bentuk angka, huruf, symbol, teks, gambar, bunyi, atau kombinasinya.

Sebagai satu kesatuan istilah, Basis Data (*Database*) sendiri dapat didefenisikan dalam sejumlah sudut pandang seperti [5] :

a. Himpunan kelompok data (arsip) yang saling berhubungan yang di organisasi sedemikian rupa agar kelak dapat dimanfaatkan kembali dengan cepat dan mudah.
b. Kumpulan data yang saling berhubungan yang disimpan secara bersamaaan sedemikian rupa dan tanpa pengulangan (redudansi) yang tidak perlu, untuk memenuhi berbagai kebutuhan.
c. Kumpulan file/ tabel/ arsip yang saling berhubungan yang disimpan dalam media penyimpan elektronik.

2.2.2.1 Operasi Dasar Basis Data

Didalam sebuah *disk*, basis data dapat diciptakan dan dapat pula ditiadakan. Didalam sebuah *disk*, kita dapat pula menempatkan beberapa (lebih dari satu) basisdata. Sementara dalam sebuah basis data, kita dapat menempatkan satu atau lebih *file*/tabel. Pada *file*/tabel inilah sesungguhnya data disimpan/ditempatkan. Setiap basis data umumnya dibuat untuk mewakili sebuah semesta data yang spesifik. Misalnya, ada basis data kepegawaian, basis data akademik, basis data inventori (Pergudangan), dan sebagainya. Sementara dalam basis data akademik, misalnya, kita dapat menempatkan *file* mahasiswa, *file* mata_kuliah, *file* dosen, *file* jadwal, *file* kehadiran, *file* nilai, dan seterusnya.

Karena itu, operasi-operasi dasar yang dapat kita lakukan berkenaan dengan basis data dapat meliputi [5] :

1. Pembuatan basis data baru (*create database*), yang identik dengan pembuatanlemari arsip yang baru.
2. Penghapusan basis data (*drop database*), yang identik dengan perusakan lemari arsip (sekaligus beserta isinya, jika ada).
3. Pembuatan *file*/tabel dari suatu basis data *(create table*), yang identik dengan penambahan map arsip baru ke sebuah lemari sarsip yang telah ada.
4. Penghapusan *file*/tabel dari suatu basis data (*drop table*), yang identik dengan perusakan map arsip lama yang ada di sebuah lemari arsip.

5. Penambahan/pengisian data baru ke sebuah *file*/tabel disebuah basis data *(insert*), yang identik dengan penambahan ke lemari arsip ke sebuah map arsip.
6. Pengambilan data dari sebuah *file*/tabel (*retrieve/search*) yang identik dengan pencarian lembaran arsip dari sebuah map arsip.
7. Pengubahan data dari sebuah *file*/tabel (*update*), yang identik dengan perbaikan isi lembaran arsip yang ada di sebuah map arsip.
8. Penghapusan data dari sebuah *file*/tabel *(delete*), yang identik dengan penghapusan sebuah lembaran arsip yang ada di sebuah map arsip. Operasi yang berkenaan dengan pembuatan objek (basis data dan tabel) merupakan operasi awal yang hanya dilakukan sekali dan berlaku seterusnya. Sedang operasi-operasi yang berkaitan dengan isi tabel (data) merupakan operasi rutin yang akan berlangsung berulang-ulang dan karena itu operasi-operasi inilah yang lebih tepat mewakili aktivitas pengelolaan (*management*) dan pengolahan (*processing*) data dalam basis data [5].

2.2.2.2 Objektif Basis Data

Telah disebutkan di awal bahwa tujuan awal dan utama dalam pengelolaan data dalam sebuah basis data adalah agar kita dapat memperoleh/menemukan kembali data (yang kita cari) dengan mudah dan cepat. Disamping itu, pemamfaatan basis data untuk pengelolaan data, juga memiliki tujuan-tujuan lain [5].

Secara lebih lengkap, pemanfaatan basis data dilakukan untuk memenuhi sejumlah tujuan (objektif) seperti buku ini [5] :

1. Kecepatan dan Kemudahan (*Speed*) Pemanfaatan basis data memungkinkan kita untuk dapat menyimpan data atau melakukan perubahan/manipulasi terhadap data atau menampilkan kembali data tersebut dengan lebih cepat dan mudah, dari pada jika kita menyimpan data secara manual (non elektronis) atau secara elektronis (tetapi tidak dalam bentuk penerapan basis data, misalnya dalam bentuk spread sheet atau dokumen teks biasa).

2. Efisiensi Ruang Penyimpanan (*Space*) Karena keterkaitan yang erat antar kelompok data dalam sebuah basis data, maka redundansi (pengulangan) data pasti akan selalu ada. Banyaknya redundansi ini tentu akan memperbesar ruang penyimpanan (baik di memori utama maupun memori sekunder) yang harus disediakan. Dengan basis data, efisiensi/optimalisasi penggunaan ruang penyimpanan dapat dilakukan, karena kita dapat melakukan penekanan jumlah redundansi data, baik dengan menerapkan sejumlah pengkodean atau dengan membuat relasi-relasi (dalam bentuk *file*) antar kelompok data yang saling berhubungan.
3. Keakuratan (*Accuracy*) Pemanfaatan pengkodean atau pembentukan relasi antar data bersama dengan penerapan aturan/batasan (*constraint*) tipe data, domain data, keunikan data, dan sebagainya, yang seara ketat dapat diterapkan dalam sebuah basis data, sangat berguna untuk menekan ketidakakuratan pemasukan/penyimpanan data.
4. Ketersediaan (*Availability*) Pertumbuhan data (baik dari sisi jumlah maupun jenisnya) sejalan dengan waktu akan semakin membutuhkan ruang penyimpanan yang besar. Padahal tidak semua data itu selalu kita gunakan/butuhkan. Karena itu kita dapat memilah adanya data utama/master/referensi, data transaksi, data histori hingga data kadarluarsa. Data yang sudah jarang atau bahkan tidak pernah lagi kita gunakan, dapat kita atur untuk dilepaskan dari sistem basis data yang sedang aktif (menjadi *off-line*) baik dengan cara penghapusan atau dengan memindahkannya ke media penyimpanan *off-line* (seperti *removable disk* atau tape). Di sisi lain, karena kepentingan pemakaian data, sebuah basis data dapat memiliki data yang disebar di banyak lokasi geografis. Data nasabah sebuah bank, misalnya, dipisah-pisah dan disimpan di lokasi yang sesuai dengan keberadaan nasabah. Dengan pemanfaatan teknologi jaringan komputer, data yang berada di suatu lokasi/cabang, dapat juga diakses (menjadi tersedia/*available*) bagi lokasi/cabang lain.
5. Kelengkapan (*Completenes*) Lengkap/tidaknya data yang kita kelola dalam sebuah basis data bersifat relatif (baik terhadap kebutuhan pemakai maupun

terhadap waktu). Bila seorang pemakai sudah menganggap bahwa data yang dipelihara sudah lengkap, maka pemakai yang lain belum tentu berpendapat sama. Atau, yang sekarang dianggap sudah lengkap, belum tentu di masa yang akan datang juga demikian. Dalam sebuah basis data, di samping data kita juga harus menyimpan struktur (baik yang mendefinisikan objek-objek dalam basis data maupun definisi dari tiap objek, seperti struktur *file*/tabel atau indeks). Untuk mengakomodasi kebutuhan kelengkapan data yang semakin berkembang, maka kita tidak hanya dapat menambah *record- record* data, tetapi juga dapat melakukan perubahan struktur dalam basis data, baik dalam bentuk penambahan objek baru (tabel) atau dengan penambahan *field-field* baru pada suatu tabel.

6. Keamanan (*Security*) Memang ada sejumlah (aplikasi) pengelola basis data yang tidak menerapkan aspek keamanan dalam penggunaan basis data. Tetapi untuk sistem yang besar dan serius, aspek keamanan juga dapat diterapkan dengan ketat. Dengan begitu kita dapat menentukan siapa-siapa (pemakai) yang boleh menggunakan basis data beserta objek-objek di dalamnya dan menentukan jenis-jenis operasi apa saja yang boleh dilakukannya.
7. Kebersamaan Pemakaian (*Sharability)* Pemakai basis data seringkali tidak terbatas pada satu pemakai saja, atau di satu lokasi saja atau oleh satu sistem/aplikasi saja. Data pegawai dalam basis data kepegawaian, misalnya, dapat digunakan oleh banyak pemakai, dari sejumlah departemen dalam perusahaan atau oleh banyak sistem (sistem penggajian, sistem akuntansi, sistem inventori, dan sebagainya). Basis data yang dikelola oleh sistem (aplikasi) yang mendukung lingkungan *multiuser*, akan dapat memenuhi kebutuhan ini, tetapi tetap dengan menjaga/menghindari (karena data yang sama diubah oleh banyak pemakai pada saat yang bersamaan) atau kondisi *deadlock* (karena ada banyak pemakai yang saling menunggu untuk menggunakan data).

2.2.3 *Database Management System*

Kumpulan atau gabungan *database* dengan perangkat lunak aplikasi yang berbasis *database* tersebut dinamakan *Database Management System* (DBMS). DBMS merupakan koleksi terpadu dari *database* dan program–program komputer (utilitas) yang digunakan untuk mengakses dan memelihara *database*. Program-program tersebut menyediakan berbagai fasilitas operasi untuk memasukan, melacak, dan memodifikasi data kedalam *database*, mendefinisikan data baru, serta mengolah data menjadi informasi yang dibutuhkan (DBMS = *Database* + *Program Utilitas*) [3]. Perangkat lunak yang termasuk DBMS seperti *dBase, FoxBase, Rbase, Microsoft-Access* (sering juga disingkat *Ms-Access*) dan Borland Pradox (untuk DBMS yang sederhana) atau *Borland-Interbase, MS-Sql, Sever, Oracle Database, IBM, DB2, Informix, Sybase, MySql, PostgreSQL* (untuk DBMS yang lebih kompleks dan lengkap) [5].

2.2.4 *Data mining*

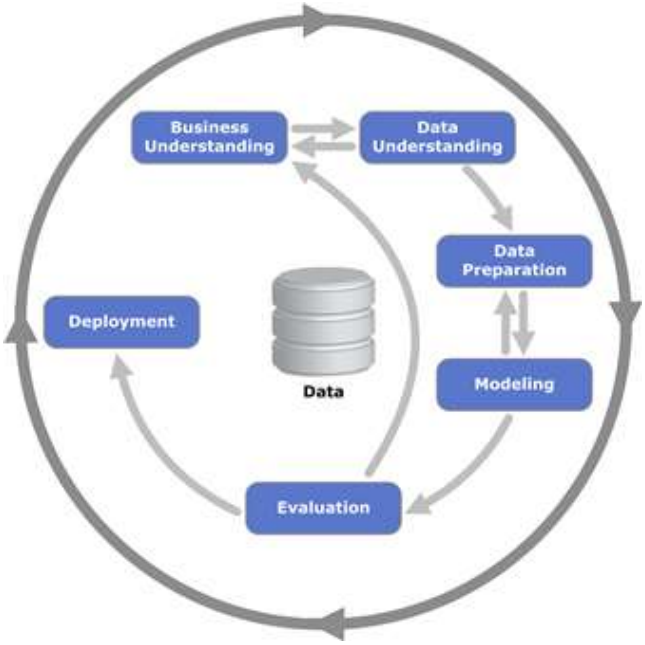
Data mining, sering juga disebut *knowledge discovery in database* (KDD), adalah kegiatan yang meliputi pengumpulan, pemakaian data historis untuk menemukan keteraturan, pola atau hubungan dalam set data berukuran besar. Keluaran dari *data mining* ini bisa dipakai untuk memperbaiki pengambilan keputusan di masa depan. Sehingga istilah *pattern recognition* sekarang jarang digunakan karena ia termasuk bagian dari *data mining* [6]. *Data mining* adalah suatu istilah yang digunakan untuk menguraikan penemuan pengetahuan didalam *database*. *Data mining* adalah proses yang menggunakan teknik statistik, matematika, kecerdasan buatan, dan *machine learning* untuk mengektrasi dan mengidentifikasi informasi yang bermanfaat dan pengetahuan yang terkait dari berbagai *database* besar (Turban, dkk. 2005) [7].

2.2.4.1 Tahapan-Tahapan *Data mining*

CRISP-DM (*CRoss-Industry Standard Process for Data mining*) merupakan suatu konsorsium perusahaan yang didirikan oleh Komisi Eropa pada tahun 1996

dan telah ditetapkan sebagai proses standar dalam *data mining* yang dapat diaplikasikan di berbagai sektor industri[1].

Penjelasan tentang siklus hidup pengembangan *data mining* yang telah ditetapkan dalam CRISP-DM diacu pada gambar 2.3.



Gambar 2. 3 Metode CRISP-DM

Berikut ini adalah penjelasan mengenai enam tahap siklus hidup pengembangan *data mining* berdasarkan gambar di atas :

1. *Business Understanding* Tahap pertama adalah memahami tujuan dan kebutuhan dari sudut pandang bisnis, kemudian menterjemakan pengetahuan ini ke dalam pendefinisian masalah dalam *data mining*. Selanjutnya akan ditentukan rencana dan strategi untuk mencapai tujuan tersebut.
2. Data *Understanding* Tahap ini dimulai dengan pengumpulan data yang kemudian akan dilanjutkan dengan proses untuk mendapatkan pemahaman yang mendalam tentang data, mengidentifikasi masalah kualitas data, atau untuk mendeteksi adanya bagian yang menarik dari data yang dapat digunakan untuk hipotesa untuk informasi yang tersembunyi.
3. Data *Preparation* Tahap ini meliputi semua kegiatan untuk membangun dataset akhir (data yang akan diproses pada tahap pemodelan/modeling) dari

data mentah. Tahap ini dapat diulang beberapa kali. Pada tahap ini juga mencakup pemilihan tabel, record, dan atribut-atribut data, termasuh proses pembersihan dan transformasi data untuk kemudian dijadikan masukan dalam tahap pemodelan (modeling).

4. Modeling Dalam tahap ini akan dilakukan pemilihan dan penerapan berbagai teknik pemodelan dan beberapa parameternya akan disesuaikan untuk mendapatkan nilai yang optimal. Secara khusus, ada beberapa teknik berbeda yang dapat diterapkan untuk masalah *data mining* yang sama. Di pihak lain ada teknik pemodelan yang membutuhan format data khusus. Sehingga pada tahap ini masih memungkinan kembali ke tahap sebelumnya.
5. *Evaluation* Pada tahap ini, model sudah terbentuk dan diharapkan memiliki kualitas baik jika dilihat dari sudut pandang analisa data. Pada tahap ini akan dilakukan evaluasi terhadap keefektifan dan kualitas model sebelum digunakan dan menentukan apakah model dapat mencapat tujuan yang ditetapkan pada fase awal (*Business Understanding*). Kunci dari tahap ini adalah menentukan apakah ada masalah bisnis yang belum dipertimbangkan. Di akhir dari tahap ini harus ditentukan penggunaan hasil proses *data mining*.
6. *Deployment* Pada tahap ini, pengetahuan atau informasi yang telah diperoleh akan diatur dan dipresentasikan dalam bentuk khusus sehingga dapat digunakan oleh pengguna. Tahap *deployment* dapat berupa pembuatan laporan sederhana atau mengimplementasikan proses *data mining* yang berulang dalam perusahaan. Dalam banyak kasus, tahap deployment melibatkan konsumen, di samping analis data, karena sangat penting bagi konsumen untuk memahami tindakan apa yang harus dilakukan untuk menggunakan model yang telah dibuat.

2.2.4.2 Metode *Data mining*

Secara garis besar, Han dalam bukunya menjelaskan bahwa metode *data mining* dapat dilihat dari dua sudut pandang pendekatan yang berbeda, yaitu pendekatan deskriptif dan pendekatan prediktif [8]. Pendekatan deskriptif adalah

pendekatan dengan cara mendeskripsikan data inputan. Metode yang termasuk kedalam pendekatan ini adalah :

1. Metode deskripsi konsep/kelas, yaitu data dapat diasosiasikan dengan kelas atau konsep. Ada tiga macam pendeskripsian yaitu (1) karakteristik data, dengan membuat *summary* karakter umum atau fitur data suatu kelas target, (2) diskriminasi data, dengan membandingkan class target dengan satu atau sekelompok kelas pembanding, (3) gabungkan antara karakterisasi dan diskriminasi.
2. Metode *association rule*, yaitu menemukan aturan asosiatif atau pola kombinasi dari suatu item yang sering terjadi dalam sebuah data.

Pendekatan kedua adalah pendekatan prediktif, yaitu pendekatan yang dapat digunakan untuk memprediksi, dengan hasil berupa kelas atau *cluster*. Metode yang termasuk dalam pendekatan ini adalah :

1. Metode klasifikasi dan prediksi, yaitu metode analisis data yang digunakan untuk membentuk model yang mendeskripsikan kelas data yang penting, atau model yang memprediksikan trend data. Klasifikasi digunakan untuk memprediksi kelas data yang bersifat kategorial, sedangkan prediksi untuk memodelkan fungsi yang mempunyai nilai kontinu.
2. Metode *clustering*, mengelompokan data untuk membentuk kelas-kelas baru atau sering disebut *cluster*. Metode clustering bertujuan untuk memaksimalkan persamaan dalam satu cluster dan meminimalkan perbedaan antar *cluster*.

2.2.5 Metode Association Rule

Association Rule atau Aturan Asosiasi adalah teknik *Data mining* untuk menemukan aturan asosiatif atau pola kombinasi dari suatu item. Bila kita mengambil contoh aturan asosiatif dalam suatu transaksi pembelian barang di suatu minimarket adalah kita dapat mengetahui berapa besar kemungkinan seorang konsumen membeli suatu item bersamaan dengan item lainnya (membeli roti bersama dengan selai). Karena awalnya berasal dari studi tentang *database*

transaksi pelanggan untuk menentukan kebiasaan suatu barang dibeli bersama apa, maka association rule sering juga dinamakan market basket analysis [6].

Association Rule adalah bentuk jika "kejadian sebelumnya" kemudian "konsekuensinya" (*If antecedent, then consequent*), yang diikuti dengan perhitungan aturan *support* dan *confidence*. Bentuk umum dari *association rule* adalah *Antecedent -> Consequent.* Bila kita ambil contoh dalam sebuah transaksi pembelian barang di sebuah minimarket didapat bentuk *association rule* roti -> selai. Yang artinya bahwa pelanggan yang membeli roti ada kemungkinan pelanggan tersebut juga akan membeli selai, dimana tidak ada batasan dalam jumlah item-item pada bagian *antecedent* ataupun *consequent* dalam sebuah rule [9].

Association rule memiliki dua tahap pengerjaan, yaitu [10]:

1. Mencari kombinasi yang paling sering terjadi dari suatu *itemset*.
2. Mendefinisikan *Condition* dan *Result* (untuk *conditional association rule*).

Dalam menentukan suatu association rule, terdapat suatu interestingness measure (ukuran kepercayaan) yang didapat dari hasil pengolahan data dengan perhitungan tertentu. Umumnya ada dua ukuran, yaitu :

1. *Support* : suatu ukuran yang menunjukkan seberapa besar tingkat dominasi suatu item/itemset dari keseluruhan transaksi. Ukuran ini menentukan apakah suatu item/*itemset* layak untuk dicari *confidence*-nya (misal, dari keseluruhan transaksi yang ada, seberapa besar tingkat dominasi suatu item yang menunjukkan bahwa item A dan item B dibeli bersamaan).
2. *Confidence* : suatu ukuran yang menunjukkan 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. Bila suatu pola memenuhi kedua nilai *minimum* parameter yang sudah ditentukan sebelumnya, maka pola tersebut dapat disebut sebagai interesting rule atau strong rule [9].

2.2.5.1 Metodologi Dasar Analisis Asosiasi

Metodologi dasar *Association Rule* terbagi menjadi dua tahap, yaitu [11]:

a. Analisa pola frekuensi tinggi Tahap ini mencari pola item yang memenuhi syarat *minimum* dari nilai *support* dalam *database.* Menurut Larose, kita bebas menentukan nilai *minimum support* (minsup) dan minimum *confidence* (mincof) sesuai kebutuhan [12]. Sebagai contoh, bila ingin menemukan data-data yang memiliki hubungan asosiasi yang kuat, minsup dan mincof-nya bisa diberi nilai yang tinggi. Sebaliknya, bila ingin melihat banyaknya variasi data tanpa terlalu mempedulikan kuat atau tidaknya hubungan asosiasi antara item-nya, nilai minsup dan mincofnya dapat diisi rendah [9].

   Untuk rekomendasi dalam menentukan *minimum support* dapat diambil dari perhitungan rata-rata 1 jenis barang pada data yang digunakan, seperti rumus berikut :

$$\textit{Minimum support} = \frac{\text{Jumlah Data Yang di Gunakan}}{\text{Jumlah Jenis Item Yang Di Gunakan}} \quad \text{(Persamaan 2-1)}$$

Nilai *support* sebuah item diperoleh dengan rumus :

$$\textit{Support}\ (A) = \frac{\text{Jumlah Transaksi Yang Mengandung A}}{\text{Jumlah Total Transaksi}} \quad \text{(Persamaan 2-2)}$$

Persamaan 2 menjelaskan bahwa nilai *support* didapat dengan cara membagi jumlah transaksi yang mengandung item A (satu item) dengan jumlah total seluruh transaksi. Sedangkan untuk mencari nilai *support* dari 2 item menggunakan rumus berikut :

$$\textit{Support}(A,B) = P(A \cap B) = \frac{\text{Jumlah Transaksi Yang Mengandung Item A\&B}}{\text{Jumlah Total Transaksi}} \quad \text{(Persamaan 2-3)}$$

Persamaan 3 menjelaskan bahwa nilai *support* 2-itemsets didapat dengan cara membagi jumlah transaksi yang mengandung item A dan item B (item pertama bersamaan dengan item yang lain) dengan jumlah total seluruh transaksi.

b. Pembentukan Aturan *Assosiatif*

Setelah semua pola frekuensi tinggi ditemukan, kemudian mencari aturan asosiatif yang memenuhi syarat *minimum* untuk *confidence* dengan menghitung confidence aturan assosiatif A -> B dari *support* pola frekuensi tinggi A dan B, menggunakan rumus :

$$Confidence = (A \rightarrow B) = \frac{\text{Jumlah Transaksi Yang Mengandung A\&B}}{\text{Jumlah Transaksi Yang Mengandung A}} \times 100\% \quad \text{(Persamaan 2-4)}$$

Persamaan 4 menjelaskan bahwa nilai *confidence* diperoleh dengan cara membagi jumlah transaksi yang mengandung item A dan item B (item pertama bersamaan dengan item yang lain) dengan jumlah transaksi yang mengandung item A (item Pertama atau item yang ada disebelah kiri).

2.2.6 Algoritma *CT-Pro*

Algoritma ini merupakan pengembangan dari algoritma *FP-GROWTH* dengan melakukan modifikasi pada tree yang digunakan. Algoritma ini menggunakan struktur *Compressed FP-Tree (CFP-Tree*) dimana informasi dari sebuah *FP-Tree* diringkas dengan struktur yang lebih kecil atau ringan, sehingga baik pembentukan tree maupun *frequent itemset* mining yang dilakukan menjadi lebih cepat. [13].

Langkah-langkah algoritma *CT-PRO* adalah sebagai berikut [14]:

1. Menemukan item-item yang *frequent*

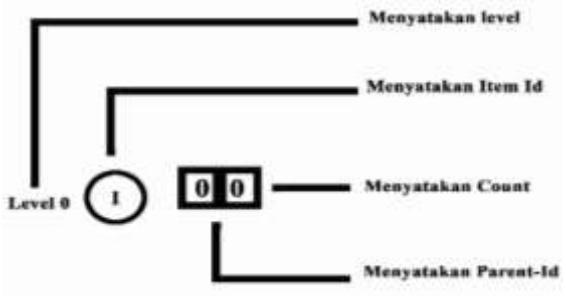
a. Data yang telah dikumpulkan, diseleksi dan pilih data yang relevan (data yang lengkap).
b. Data yang ada, kemudian dilakukan transformasi data.
c. Kemudian masing masing data diseleksi berdasarkan *minimum support* yang telah ditentukan, kemudian didapat *Item Frequent Table*.

d. Masing-masing item dihitung frekuensi kemunculannya sehingga dihasilkan *global item table*.
e. Data kemudian dimapping berdasarkan *index* pada *global item table*.

2. Membuat *CFP-Tree*

Setelah ditemukan item-item yang *frequent* kemudian dilakukan pembangunan *CFP-Tree*. *Frequent item* yang ada diurutkan sesuai global item dari nilai yang terbesar ke terkecil. *CFP-Tree* adalah tree dengan properti sebagai berikut :

a. *CFP-Tree* terdiri dari tree yang memiliki root yang mewakili *index* dari item dengan tingkat kemunculan tertinggi dan kumpulan subtree sebagai anak dari root.
b. Jika I = {i1,i2, …, ik} adalah kumpulan dari *frequent item* dalam transaksi, item dalam transaksi akan dimasukkan kedalam *CFP-Tree* dimulai dari root subtree yang merupakan i1 dalam header table.
c. Root dari *CFP-Tree* merupakan level-0 dari tree.
d. Setiap node dalam *CFP-Tree* memiliki empat *field* utama yakni item-id, *parentid*, count yang merupakan jumlah item pada node tersebut, dan level yang menunjukkan struktur data tree pada node tersebut dimulai dari item yang terdapat pada *header* table dengan level yang terdapat pada *CFP- Tree* [13].



Gambar 2. 4 Struktur *CFP-Tree*

3. Melakukan penggalian *frequent patterns*

Setelah tahap pembangunan *CFP-Tree* dari sekumpulan data transaksi, akan diterapkan algoritma *CT-Pro* untuk mencari *frequent itemset* yang signifikan. Berikut adalah langkah-langkah dari algoritma *CT-Pro* :

a. Lakukan pencarian node yang berkaitan dengan item dimana pencarian dimulai dari global item dengan *support count* terkecil sampai *global item* dengan *support count* terbesar karena *CT-Pro* bekerja *Bottom-Up* pada *Global CFP-Tree*.
b. Dari semua node yang ditemukan untuk setiap item inilah yang disebut dengan *Local Frequent* item dan digunakan untuk membuat *Local Item Table* yang pembuatannya dilakukan berdasarkan jumalah *minimun support* yang telah ditentukan.
c. Selanjutnya dibuat *Local CFP-Tree* berdasarkan *Local Item Table* yang terbentuk.
d. Dari *Local CFP-Tree* yang ada, kita dapat membuat *frequent patternnya*. Dari hasil *frequent pattern* ini akan dilakukan mining dengan rumus *confidence*.

2.2.7 Unified Modelling Language (UML)

UML singkatan dari *Unified Modeling Languages* yang berarti bahasa pemodelan standar. Ketika kita membuat model menggunakan konsep UML ada aturan-aturan yang harus diikuti. Bagaimana elemen pada model-model yang kita buat berhubungan satu dengan yang lainnya harus mengikuti standar yang ada. UML bukan hanya sekedar diagram tetapi juga menceritakan konteksnya [15].

Berikut adalah beberapa model yang digunakan dalam perancangan *Data mining* pemaketan barang di Mini Market Barokah untuk menggambarkan sistem dalam UML:

1. Diagram *Use Case*
2. Diagram *Activity*
3. Diagram *Sequens*ce
4. Diagram *Class*

2.2.7.1 *Use Case Diagram*

Diagram *Use Case* menggambarkan apa saja aktifitas yang dilakukan oleh suatu sistem. *Use Case* menggambarkan fungsi tertentu dalam suatu sistem berupa komponen, kejadian atau kelas. Komponen Pembentuk *Use Case* Diagram adalah sebagai berikut :

1. *Actor*

Pada dasarnya *actor* bukanlah bagian dari use case diagram, namun untuk dapat terciptanya suatu *use case* diagram diperlukan beberapa *actor*. Actor tersebut mempresentasikan seseorang atau sesuatu (seperti perangkat, sistem lain) yang berinteraksi dengan sistem. Sebuah *actor* mungkin hanya memberikan informasi inputan pada sistem, hanya menerima informasi dari sistem atau keduanya menerima, dan memberi informasi pada sistem. *Actor* hanya berinteraksi dengan *use case*, tetapi tidak memiliki kontrol atas *use case*. *Actor* digambarkan dengan *stick man* . *Actor* dapat digambarkan secara secara umum atau spesifik, dimana untuk membedakannya kita dapat menggunakan *relationship*.

2. *Use Case*

Use case adalah gambaran fungsionalitas dari suatu sistem, sehingga *customer* atau pengguna sistem paham dan mengerti mengenai kegunaan sistem yang akan dibangun.

Catatan : *Use case* diagram adalah penggambaran sistem dari sudut pandang pengguna sistem tersebut (*user*), sehingga pembuatan *use case* lebih dititikberatkan pada fungsionalitas yang ada pada sistem, bukan berdasarkan alur atau urutan kejadian.

Cara menentukan *Use Case* dalam suatu sistem:

a. Pola perilaku perangkat lunak aplikasi.
b. Gambaran tugas dari sebuah *actor*.
c. Sistem atau “benda” yang memberikan sesuatu yang bernilai kepada *actor*.

d. Apa yang dikerjakan oleh suatu perangkat lunak (*bukan bagaimana cara mengerjakannya).

Ada beberapa relasi yang terdapat pada *use case diagram*:

a. *Association*, menghubungkan *link* antar *element*.
b. *Generalization*, disebut juga *inheritance* (pewarisan), sebuah elemen dapat merupakan spesialisasi dari elemen lainnya.
c. *Dependency*, sebuah element bergantung dalam beberapa cara ke element lainnya.
d. *Aggregation*, bentuk assosiation dimana sebuah elemen berisi elemen lainnya.

Tipe relasi/ *stereoType* yang mungkin terjadi pada *use case* diagram:

a. <<*include*>> , yaitu kelakuan yang harus terpenuhi agar sebuah *event* dapat terjadi, dimana pada kondisi ini sebuah use case adalah bagian dari *use case* lainnya.
b. <<*extends*>>, kelakuan yang hanya berjalan di bawah kondisi tertentu seperti menggerakkan alarm.
c. <<*communicates*>>, mungkin ditambahkan untuk asosiasi yang menunjukkan asosiasinya adalah communicates association . Ini merupakan pilihan selama asosiasi hanya tipe *relationship* yang dibolehkan antara *actor* dan *use case*.

2.2.7.2 *Activity Diagram*

Activity diagram memiliki pengertian yaitu lebih fokus kepada menggambarkan proses bisnis dan urutan aktivitas dalam sebuah proses. Dipakai pada *business* modeling untuk memperlihatkan urutan aktifitas proses bisnis. Memiliki struktur diagram yang mirip *flowchart* atau data *flow* diagram pada perancangan terstruktur. Memiliki pula manfaat yaitu apabila kita membuat diagram ini terlebih dahulu dalam memodelkan sebuah proses untuk membantu memahami proses secara keseluruhan. Dan *activity* dibuat berdasarkan sebuah atau beberapa *use case* pada *use case* diagram.

2.2.7.3 *Squence Diagram*

Diagram *sequen* menggambarkan interaksi objek pada *use case* dengan mendeksripsikan waktu hidup objek dan pesan yang dikirimkan dan di terima antar objek. Oleh karena itu untuk menggambarkan diagram *sequen* maka harus diketahui objek-objek yang terlibat dalam sebuah use case beserta metode-metode sekuen juga di butuhkan untuk melihat skenario yang ada pada *use case*. *Sequens*ce diagram biasa digunakan untuk menggambarkan skenario atau rangkaian langkah- langkah yang dilakukan sebagai *respons* dari sebuah *event* untuk menghasilkan *output* tertentu. Diawali dari apa yang men-trigger aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara internal dan output apa yang dihasilkan. Masing-masing objek, termasuk aktor, memiliki *lifeline* vertikal. *Message* digambarkan sebagai garis berpanah dari satu objek ke objek lainnya.

2.2.7.4 *Class Diagram*

Diagram kelas atau *class diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan di buat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan metode atau oprasi.

a. Atribut merupakan variabel-variabel yang di miliki oleh suatu kelas.
b. Oprasi 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. banyak berbagai kasus, perancangan kelas yang dibuat tidak sesuai dengan kelas-kelas yang dibuat pada perangkat lunak, sehingga tidaklah ada gunanya lagi sebuah perancangan karena apa yang dirancang dan hasil jadinya tidak sesuai.