BAB II
TINJAUAN PUSTAKA

II.1 Tinjauan Perusahaan

CV. Aldo Putra berlokasi di Jalan Pasar Induk Gedebage No. 89/104 Bandung, bergerak dibidang grosir pakaian jadi impor. Barang yang dijual di CV. Aldo Putra dijual per karung dan setiap karungnya berisi satu jenis barang. Barang-barang tersebut disuplai oleh banyak perusahaan dari luar negeri seperti Samurai, HPL, Kamerun, Hongyang, dan lain-lain. Pelanggan yang memesan barang di CV. Aldo Putra berasal dari berbagai wilayah di seluruh Indonesia, seperti Papua, Sumatera, Bali dan wilayah lainnya yang terdiri dari berbagai toko dalam partai kecil atau besar.

II.1.1 Sejarah Instansi

CV. Aldo Putra berdiri pada tahun 1998 di pasar Tegallega. Saat itu CV. Aldo Putra bernama Putra Ardes dan menggunakan cara manual dalam mencatat semua transaksi yaitu masih menggunakan buku dan nota biasa. Berdasarkan surat Wali Kota Bandung 21 Juli 2004,pedagang Tegallega dipindah ke Pasar Induk Gedebage. Sejak pindah ke Pasar Induk Gedebage pada akhir 2004, pemilik perusahaan berusaha memperbaiki kinerja karyawannya yaitu menambahkan satu unit komputer,uang untuk membeli komputer tersebut berasal dari uang kompensasi pemerintah saat itu.Komputer tersebut difungsikan untuk mengolah data transaksi yang tadinya masih menggunakan buku dan nota biasa. Menurut pihak perusahaan data yang disimpan di komputer kemungkinan untuk rusak dan hilang sangat kecil, jadi sejak saat itu perusahaan mempunyai dua cara untuk mencatat data transaksi. Pada tahun 2012 CV. Aldo Putra resmi berbadan hukum dengan nama CV. ALDO PUTRA.

II.1.2 Struktur Organisasi

Struktur organisasi adalah pola hubungan antara bagian-bagian dari instansi atau menggambarkan dengan jelas pemisahan kegiatan pekerjaan antara bagian yang satu dengan bagian yang lain dalam suatu instansi. Gambar II-1berikut merupakan struktur organisasi yang ada CV. Aldo putra :



Gambar II-1 Struktur Organisasi

II.1.3 Deskripsi Kerja Struktur Organisasi

Adapun deskripsi kerja dari struktur organisasi di CV. Aldo Putra adalah :

1. Direktur

   Pemilik perusahaan yang bertanggung jawab kepada perusahaan dan karyawan yang bekerja di CV. Aldo Putra.

2. Sekretaris

   Membantu pimpinan mengerjakan tugas kecil pimpinan, seperti menerima surat masuk, menangani janji, menangani telepon.

3. Supervisor

   Bagian yang menangani orang – orang yang memproduksi dan atau melakukan kinerja pelayanan. Dalam hal ini mengepalai bagian gudang dan administrasi. Supervisor juga melakukan analisa terhadap pengeluaran dan pengadaan barang.

4. Administrasi

   Bagian yang menangaini pelayanan, pencatatan dan transaksi yang terjadi di CV. Aldo Putra.

5. Bagian Gudang

   Bagian yang menangani penyetokan barang, pencatatan barang masuk dan barang keluar serta mengantarkan barang ke tempat pelanggan jika diperlukan.

II.2 Landasan Teori

Sub bab ini berisi teori-teori pendukung yang digunakan dalam proses analisis dan implementasi dalam tugas akhir ini.

II.2.1 Pengertian Data

Data adalah rekaman mengenai fenomena atau fakta yang ada atau yang terjadi.

"Menurut Fatansyah, data adalah refresentasi fakta dunia nyata yang mewakili suatu objekseperti manusia (pegawai, siswa, pembeli, pelanggan), barang, hewan, peristiwa, konsep, keadaan, dan sebagainya, yang direkam dalam bentuk angka, huruf, teks, gambar, bunyi, atau kombinasinya."

Data merupakan suatu bentuk keterangan-keterangan yang belum diolah atau dimanipulasi sehingga belum begitu berarti bagi sebagian pemakai. Sedangkan informasi merupakan data yang sudah di olah sehingga memiliki arti.

II.2.2 Basis Data

"Menurut Abdul Kadir, database adalah koleksi data yang saling terkait. Dapat dianggap sebagai suatu penyusunan data terstruktur yang disimpan dalam media pengingat (hardisk) yang bertujuan agar data tersebut dapat diakses dengan mudah dan cepat."

"Menurut Fatansyah, basis data adalah himpunan kelompok data yang saling berhuungan yang diorganisasi sedemikian rupa agar kelak dapat dimanfaatkan kembali dengan cepat dan mudah."

II.2.3 Data Mining

Beberapa pengertian dari beberapa ahli tentang definisi *data mining* :

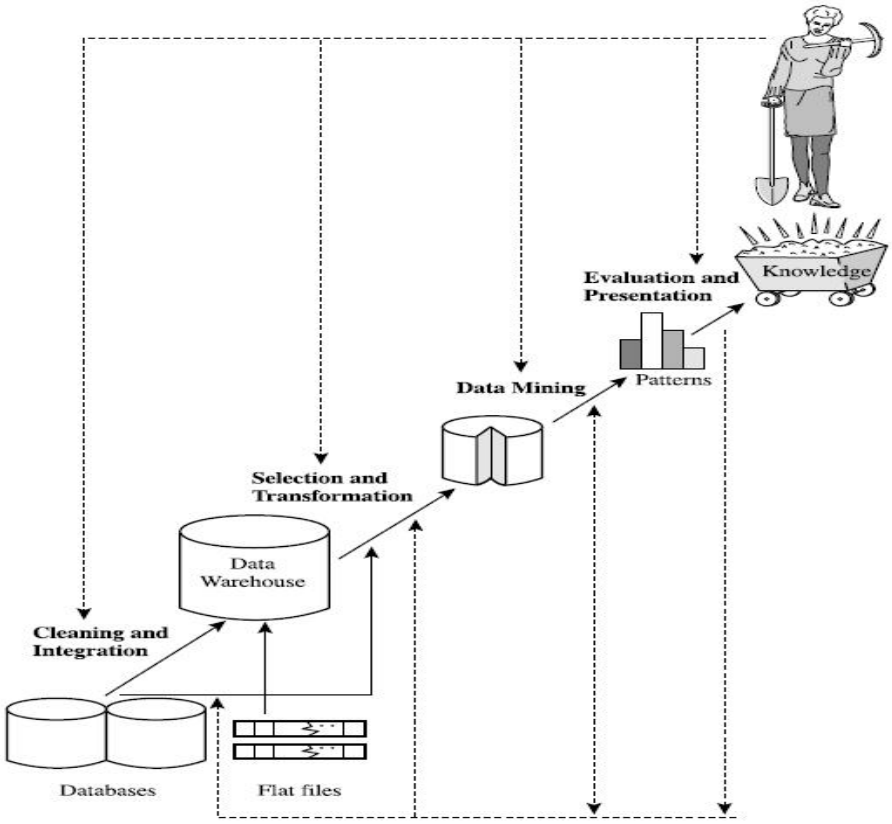
"Menurut M. Fairuzabadi, d*ata mining adalah serangkaian proses untuk menggali nilai tambah berupa informasi yang selama ini tidak diketahui secara manual dari suatu basisdata. Informasi yang dihasilkan diperoleh dengan caramengekstraksi dan mengenali pola yang penting atau menarik dari data yang terdapat dalam basisdata."*

"Menurut Jiawei Han dan Micheline Kamber, data mining adalah menarik pola atau pengetahuan dari jumlah data yang besar yang sebelumnya tidak diketahui dan berpotensi berguna."

"Menurut Turban dkk (Kusrini & Emha Taufiq, 2009),data mining adalah proses yang menggunakan teknik statistik, matematika, kecerdasan buatan, dan machine learning untuk mengekstraksi dan mengidentifikasi informasi yang bermanfaat dan pengetahuan yang terkait dari berbagai database besar."

II.2.4 Tahapan Data Mining

Istilah *data mining* yang popular saat ini dikenal sebagai *Knowledge Discovery from Data* atau KDD. *Data mining* merupakan suatu bagian langkah yang penting dalam proses penemuan pengetahuan terutama berkaitan dengan ekstraksi dan perhitungan pola-pola (Han, Jiawei; Kamber, Micheline, 2006). Seperti yang ditunjukkan pada Gambar II-2 berikut ini:



Gambar II-2 Tahapan *Data Mining*

a. *Data Cleaning*

Tahapan ini dilakukan untuk menghilangkan data *noise* dan data yang tidak konsisten dengan tujuan akhir dari proses data mining.

b. *Data Integration*

Tahapan ini dilakukan untuk menggabungkan atau mengkombinasikan dari *multiple data source*.

c. *Data Selection*

Pada tahapan ini adalah memilih atau menyeleksi data apa saja yang yang relevan dan diperlukan dari *database*.

d. *Data Transformation*

Untuk mentransformasikan data ke dalam bentuk yang lebih sesuai untuk di-*mining*.

e. *Data Mining*

Proses terpenting dimana metode tertentu diterapkan dalam *database* untuk menghasilkan *datapattern*.

f. *Pattern Evaluation*

Untuk mengidentifikasi apakah *interentingpatterns* yang didapatkan sudah cukup mewakili *knowledge* berdasarkan perhitungan tertentu.

g. *Knowladge Persentation*

Untuk mempresentasikan *knowledge* yang sudah didapatkan dari *user*.

II.2.5 Arsitektur Data Mining

Pada umumnya sistem *data mining* terdiri dari komponen–komponen berikut ini:

a. *Database*, *Data Warehouse*, *World Wide Web*, atau media penampung data lainnya

Media pada komponen ini dapat berupa *database*, *data warehouse*, *spread sheet*, atau jenis media penampung data lainnya. *Data cleaning* dan data *integration* dapat dilakukan pada data data tersebut.

b. *Database* atau *Data Warehouse Server*

Database atau *Data Warehouse Server* bertanggung jawab untuk menyediakan data yang relevan berdasarkan permintaan dari user pengguna *data mining*.

c. Knowladge base

Merupakan basis pengetahuan yang digunakan sebagai panduan dalam pencarian pola.

d. Data mining engine

Ini penting untuk system data mining idealnya terdiri dari set modul fungsional untuk tugas – tugas seperti, karakterisasi, asosiasi dan analisis korelasi, klasifikasi, prediksi, analisis klaster, analisis outlier dan analisis evolusi.

e. Patern evaluation modul

Komponen ini biasanya menggunakan langkah – langkah dan berinteraksi dengan modul – modul data mining sehingga focus dalam pencaraian pola yang menarik. Untuk data mining yang efisien, sangat dianjurkan untuk mengevaluasi penarikan pola secara mendalam sehingga membatasi hanya pada pencarian pola yang menarik.

f. User interface

Bagian ini merupakan sarana antara *user* dan sistem *data mining* untuk berkomunikasi, dimana *user* dapat berinteraksi dengan sistem melalui data mining *query*, untuk menyediakan informasi yang dapat membantu dalam pencarian *knowledge*. Selain itu, komponen ini memungkinkan penguna dalam pencarian database dan skema data warehouse atau struktur data, mengevaluasi pola *mining*, dan mengvaluasikan pola dalam bentuk yang berbeda.

Gambar II-3 dibawah ini menunjukkan arsitektur *data mining* yang telah dijelaskan diatas :



Gambar II-3 Arsitektur *Data Mining*

II.2.6 *Association Rule*

"Menurut Jiawei Han and Micheline Kamber, Association rule adalah mencari pola – pola, asosiasi, korelasi, atau struktur kausal antara set item dalam database transaksi, database relasional dan repositori dan lainnya".

"Menurut Kusrini dan Emha Taufiq Luthfi, association rule adalah teknik data mining untuk menemukan aturan asosiatif antara suatu kombinasi item."

Association rule mempunyai parameter, yaitu *support, confidence dan correlation*. Penggunaan *parameter support* dan *confidence* hanya untuk assosiasi data yang menghasilkan beberapa aturan dalam menentukan metode. Namun, kita dapat meningkatkan hasilnya dengan *parameter correlation* (Han, Jiawei; Kamber, Micheline, 2006).

Dalam *association rule*, suatu kelompok *item* dinamakan *itemset*. *Support* adalah suatu ukuran yang menunjukkan 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 menunjukkan bahwa *item* A dan item B dibeli bersamaan).*Support* dari *itemset* X adalah persentase transaksi di D yang mengandung X, biasa ditulis dengan *supp(X)*. Jika *support* suatu *itemset* lebih besar atau sama dengan *minimum support*, maka *itemset* tersebut dapat dikatakan sebagai *frequent itemset*, yang tidak memenuhi dinamakan *infrequent*. *Confidence* adalah suatu ukuran yang menunjukkan hubungan antara dua*item* (misal, menghitung kemungkinan seberapa sering item B dibeli oleh pelanggan jika pelanggan tersebut membeli sebuah item A). *Confidence* suatu *rule* R (X=>Y) adalah proporsi dari semua transaksi yang mengandung baik X maupun Y dengan yang mengandung X, biasa ditulis sebagai *conf(R)*. Sebuah *association rule* dengan *confidence* sama atau lebih besar dari minimum *confidence*Y dapat dikatakan sebagai valid *association rule*.

Aturan asosiasi biasanya dinyatakn dalam bentuk :

{roti, mentega} -> {susu} (support = 40%, confidence = 50%)

Aturan tersebut berarti, 50% dari transaksi di *database* yang memuat *item* roti dan mentega juga memuat *item* susu. Sedangkan 40% dari seluruh transaksi yang ada di database memuat ketiga *item* itu. Dapat juga diartikan, bila seorang konsumen membeli roti dan mentega punya kemungkinan 50% untuk juga membeli susu. Aturan ini cukup signifikan karena mewakili 40% dari catatan transaksi selama ini.

Metode dasar analisis asosiasi tebagi menjadi dua tahap :

1. Analisis pola frekuensi tinggi

   Tahap ini mencari kombinasi *item* yang memenuhi syarat minimum dari nilai *support* dalam *database*. Menurut Larose yang dikutif oleh Kusrini, kita bebas menentukan nilai *minimum support* (minsup) dan *minimum confidence* (mincof) sesuai kebutuhan (Kusrini; Luthfi, Emha Taufiq;, 2009). Sebagai contoh, bila ingin menemukan data-data yang memiliki hubungan asosiasi yang kuat, minsup dan mincofnya 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.

   Nilai *support* sebuah *item* diperoleh dengan rumus berikut.

$$support(A) = \frac{Jumlahtransaksimengandung A}{Totaltransaksi} \qquad \text{(II-1)}$$

   Persamaan 1 menjelaskan bahwa nilai *support* didapat dengan cara membagi jumlah transaksi yang mengandung item A (satu item) dengan jumlah total seluruh transaksi.Sementara itu, nilai *support* dari 2 item diperoleh dari rumus 2 berikut:

$$support(A,B) = P(A \cap B)$$

$$support(A,B) = \frac{\sum Transaksi\ mengandung\ A\ dan\ B}{\sum Transaksi} \qquad \text{(II-2)}$$

   Support count adalah banyaknya *itemsets* yang sama muncul secara bersamaan pada suatu data transaksi pada keranjang belanja.

2. Pembentukan Aturan Asosiasi

Setelah semua pola frekuensi tinggi ditemukan,barulah dicari *association rule* yang memenuhi syarat minimum untuk *confidence* dengan menghitung *confidence* aturan asosiatif A → Bdari support pola frekuensi tinggi A dan B.

Nilai *confidence* dari aturan A→B diperoleh dengan rumus berikut.

$$confidence = P(B|A) = \frac{\sum Transaksi\ mengandung\ A\ dan\ B}{Transaksi\ mengandung\ A} \qquad \text{(II-3)}$$

II.2.7 Parameter *Corellation*

Correlation merupakan parameter yang menetukan nilai keterhubungan. *Parameter correlation* dicari menggunakan rumus *lift*. Nilai *lift* mengukur seberapa penting *rule* yang telah terbentuk berdasarkan nilai *support* dan *confidence*. Terjadinya *itemset* A adalah independen dari terjadinya *itemset* B jika P(A|B)=P(A)P(B), jika tidak, *itemset* A dan B tergantung dan berkorelasi sebagai peristiwa.

Lift yang terjadi antara A dan B dapat diukur dengan menghitung :

$$lift(A,B) = \frac{P(A \cup B)}{P(A)P(B)} \qquad \text{(II-4)}$$

Dengan aturan sebagai berikut :

1. *Lift* (A,B) < 1 maka membeli(B)
2. *Lift* (A,B) >1 maka membeli (A,B)

Sebuah transaksi dikatakan valid jika mempunyai nilai *lift* lebih dari 1, yang berarti bahwa dalam transaksi tersebut *item*A dan *item* B benar-benar dibeli secara bersamaan.

II.2.8 Algoritma Apriori

Apriori adalah suatu algoritma yang sudah sangat dikenal dalam melakukan pencarian *frequent itemset* untuk mendapatkan *association rules.* Sesuai dengan namanya, algoritma ini menggunakan pengetahuan mengenai *frequent itemset* yang telah diketahui sebelumnya untuk memproses informasi selanjutnya. Apriori

menggunakan pendekatan secara *iterative* yang disebut juga sebagai*level-wise search* dimana k-*itemset* digunakan untuk mencari (k+1)-*itemset*.

Langkah – langkah algoritma apriori sebagai berikut :

1. Set k=1 (menunjuk pada *itemset* ke-1).
2. Hitung semua k-*itemset* (*itemset* yang mempunyai k *item*).
3. Hitung *support* dari semua calon *itemset*. *Filter itemset* tersebut berdasarkan nilai *minimum support*-nya.
4. Gabungkan semua k-*sized itemset* untuk menghasilkan calon *itemset* k+1.
5. Set k=k+1 (menunjuk pada *itemset* ke-2 dan seterusnya)
6. Ulangi langkah 3-5 sampai tidak ada *itemset* yang lebih besar yang dapat dibentuk.
7. Buat final *set* dari *itemset* dengan menciptakan suatu *union* dari semua k-*itemset*.

Untuk meningkatkan efisiensi dari pencarian k-*itemset*, maka digunakan suatu metode yang dinamakan *apriori property*, metode ini dapat mengurangi lingkup pencarian sehingga waktu pencarian dapat dipersingkat. Pengertian *apriori property* adalah suatu kondisi dimana semua *subset* dari *frequent itemset* yang tidak kosong haruslah juga *frequent*, *frequent* yang dimaksud disini adalah semua *itemset* yang memenuhi *minimum support* yang ditentukan oleh pengguna. Berdasarkan definisi yang disebutkan, jika suatu *itemset*I tidak memenuhi *minimum support* (min_sup) yang ditentukan, maka I tidak *frequent*, atau dinotasikan dengan P(I)<min_sup. Jika suatu *item* A dimasukkan kedalam I, maka hasil *itemset*I U Atidak dapat terjadi lebih dari jumlah itemset I. oleh karena itu I U A tidak *frequent* juga, dengan kata lain P(I U A)<*min_sup*. Secara sederhana dapat dikatakan jika suatu *itemset* tidak memenuhi syarat, maka semua *superset* (*itemset* lain yang dibentuk dengan mengandung unsur *itemset* tersebut) dari *itemset* tersebut tidak mungkin akan memenuhi syarat juga.

Berikut penjelasan lebih lanjut mengenai algoritma *apriori property* yang digunakan dalam algoritma ini. Penjelasan berikut merupakan proses yang terdiri dari dua langkah yaitu *join* dan *prune* (Han, Jiawei; Kamber, Micheline, 2006):

1. Langkah join : untuk menentukan L_k, suatu set kandidat k-*itemset* dihasilkan dengan cara menjoinkan L_{k-1} dengan dirinya sendiri. Set kandidat ini dihasilkan sebagai C_k. Misalnya I_1 dan I_2 adalah *itemset* di dalam $L_{k-1.}$ Sebagai contoh $I_1[k-2]$ mengacu pada kedua untuk *item* terakhir di I_1. Perlu ditekankan bahwa algoritma ini mempunyai aturan bahwa *item* di dalam transaksi atau *itemset* telah diurutkan secara *lexicographic order* terlebih dahulu. Selanjutnya *join* yang dinotasikan dengan $L_{k-1} \infty L_{k-1}$ dijalankan, dimana anggota-anggota L_{k-1} yang berupa *itemset-itemset* dapat di-*join* jika *first* (k-2)-*items* dari L_{k-1} sama. Jadi anggota l_1 dan l_2 dari L_{k-1} akan di-*join* jika $(l_1[1]=I_1[1])\wedge(I_1[2] = I_1[2]) \wedge \ldots \wedge (I_1[k-2] = I_2[k-2]) \wedge (I_1[k-1] < I_2[k-1])$. Kondisi $(I_1[k-1] < I_2[k-1])$ bertujuan untuk memastikan tidak ada duplikat *itemset* yang dihasilkan dari join I_1 dan I_2 yaitu dalam format $I_1[1], I_1[2],\ldots,I_1[k-2], I_1[k-1], L_2[k-2]$.
2. Langkah *prune* : C_k adalah *superset* dari $L_k$dimana setiap anggotanya bisa *frequent* ataupun tidak, tetapi semua *frequentk-itemset* termasuk dalam C_k. Proses *scan* terhadap *database* yang dilakukan untuk menentukan jumlah kemunculan setiap *candidate* yang ada di dalam C_k akan menentukan L_k (sebagai contoh, semua *candidate* yang memiliki *support count* tidak kurang atau sama dengan *minimum support* yang ditentukan disebut *frequent*, yang artinya juga memenuhi syarat untuk masuk menjadi L_k). Ada kemungkinan C_k dapat mengandung *candidate* dengan jumlah yang sangat besar, sehingga dapat menyebabkan proses perhitungan C_k menjadi berat. Untuk mengurangi jumlah *candidate* C_k, *apriori property* digunakan sebagai berikut.Semua (k-1) *itemset* yang tidak *frequent*

dinyatakan tidak dapat menjadi *subset* dari sebuah *frequent* k-*itemset*. Oleh karena itu, jika ada (k-1)-*subset* dari sebuah *candidate* k-*itemset* tidak termasuk dalam L_{k-1}, maka *candidate* tersebut tidak mungkin *frequent* juga dan oleh karena itu dapat di-*remove* dari C_k.

Berikut ini diberikan penjelasan lebih lanjut melalui contoh kasus, pemakaian algoritma *apriori* untuk menemukan *association rule*. Contoh tabel berikut adalah merupakan transaksi *database* D dari sebuah toko. Ada sembilan transaksi didalam *database* ini.

Table II-1 Contoh tabel *Transaction Database* D

Kode Transaksi	Item yang dibeli
T100	I1,I2,I5
T200	I2,I4
T300	I2,I3
T400	I1,I2,I4
T500	I1,I3
T600	I2,I3
T700	I1,I3
T800	I1,I2,I3,I5
T900	I1,I2,I3

Berikut adalah langkah-langkah algoritma Apriori untuk menemukan *frequent itemset*:

a. Pada iterasi dari algoritma, setiap item adalah anggota set dari *candidate* 1-*itemset*, C_1. Algoritma akan secara langsung memeriksa semua transaksi yang ada di dalam *database* D untuk dapat menghitung kejadian munculnya setiap *item*.
b. Jika diasumsikan bahwa *minimum support* yang dibutuhkan adalah 2 (dalam hal ini *min_sup*=2/9=22%). Set dari *frequent* 1-*itemset* dapat diperoleh dari semua *candidate* 1-*itemset* atau C_1 yang memenuhi

minimum *support*. Dalam contoh ini kebetulan semua anggota C_1 memiliki *support count* ≥ *minimum support*. Jadi tidak ada anggota C_1 yang di hapus pada L_1.

c. Untuk menemukan frequent 2-*itemset* atau L_2, pertama-tama algoritma ini menggunakan join $L_1 \infty L_1$ untuk menghasilkan *candidate* 2-*itemset* atau C_2.

d. Kemudian, transaksi yang ada pada *database* D diperiksa atau di-*scan* dan *support count* dari tiap *candidate itemset* yang ada di C_2 ditambahkan, seperti yang ditunjukkan pada Gambar II-4. Dapat dilihat pada kolom sebelah kanan,setiap kandidat *itemset* yang ada dalam C_2 tertera *support count* dari masing-masing *candidate* 2-*itemset*.

e. Set dari 2-*itemset* atau L_2, ditentukan dari semua *candidate* 2-*itemset* yang memenuhi *minimum support*. Dapat dilihat pada Gambar II-4bahwa *itemset* pada C_2 yang memiliki *support count* lebih kecil dari 2 dihapus sehingga yang tersisa adalah *itemset* yang memiliki *support count* ≥ *minimum support*.

f. Proses untuk menghasilkan atau men-*generate* set dari candidate 3-*itemset* atau C_3, dijelaskan secara lebih *detail* dalam dua langkah yaitu *join* dan *prune* di dalam *apriori property*.

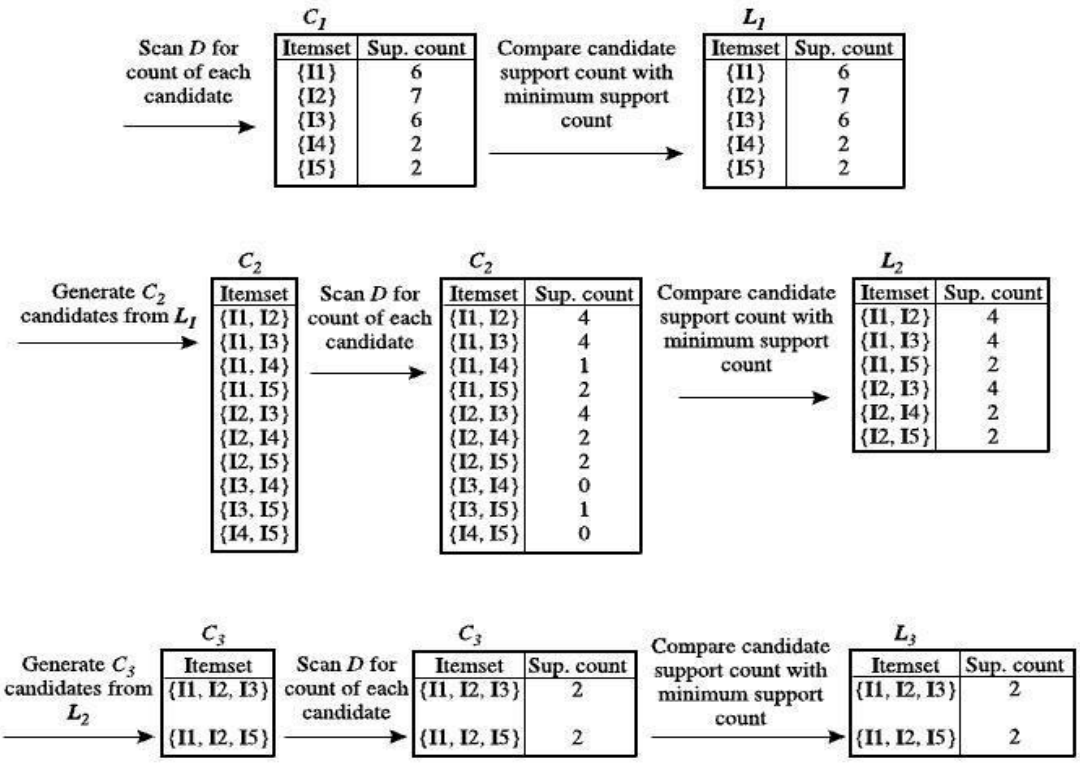
   Join step: Langkah pertama di dalam mendapatkan C_3, yaitu dengan mengkombinasikan L_2 dengan L_2 melalui proses *join* $L_2 \infty L_2$, menghasilkan $\{\{I_1,I_2,I_3\},\{I_1,I_2,I_5\}, \{I_1,I_3,I_5\}, \{I_2,I_3,I_4\}, \{I_2,I_3,I_5\}, \{I_2,I_4,I_5\}\}$. Detail dari $L_2 \infty L_2$ dapat dilihat padaGambar II-4**.**

   Prune step: Berdasarkan pada ketentuan *apriori property* yang mengatakan bahwa semua *subset* dari masing-masing *frequentitemset* diatas juga harus *frequent*, maka dapat ditentukan bahwa keempat *candidate* terakhir yaitu $\{\{I_1,I_3,I_5\}, \{I_2,I_3,I_4\}, \{I_2,I_3,I_5\}, \{I_2,I_4,I_5\}\}$ tidak mungkin *frequent*. Oleh karena itu, keempat *candidate* tadi harusdi-

remove dari C_3. Dengan demikian dapat menghemat waktu yang tidak diperlukan untuk melakukan *scan* pada *database,* ketika akan menetukan L_3. Perlu diingat, bahwa ketika diperoleh *candidate* k-*itemset* atau C_k setelah langkah *join*, maka perlu di periksa terlebih dahulu, apakah (k-1)-*subset* dari masing-masing anggota C_k adalah *frequent*, sehubungan dengan algoritma apriori menggunakan strategi *level-weis search*. Jadi *apriori property* ini dilakukan secara *level-weis search* pada L_k dimana k dimulai dari level 2 yaitu L_2 dan seterusnya.

g. Semua transaksi di dalam D diperiksa atau di-*scan* untuk menentukan L_3 yaitu terdiri dari candidate 3-*itemset* di dalam C_3 yang memenuhi *minimum support* yang sudah ditentukan.

h. Algoritma akan melakukan kombinasi antara L_3 dan L_3 untuk menghasilkan set dari *candidate* 4-*itemset*, C_4. Walaupun hasil join adalah {I1,I2,I3}, itemset ini di-prune karena salah satu (k-1)-*subset*-nya yaitu {I2,I3,I5} tidak *frequent*. Dengan demikian $C_4 = \emptyset$, dan jalannya proses algoritma ini dihentikan karena telah menemukan sebuah frequent itemset.



Gambar II-4 Pencarian *candidate dan frequent itemset* menggunakan *apriori*