Perancangan dan Pembuatan Modul Data Mining Market Basket Analysis pada Odoo dengan Studi Kasus Supermarket X

Stefani Natalia Hendratha[1], Yulia[2], Gregorius Satia Budhi[3]
Program Studi Teknik Informatika, Fakultas Teknologi Industri, Universitas Kristen Petra
Jln. Siwalankerto 121 – 131 Surabaya 60236
Telp. (031)-2983455, Fax. (031)-8417658
Email : nh.stefani@gmail.com[1], greg@petra.ac.id[2], silvia@petra.ac.id[3].

ABSTRAK

Sistem *Enterprise Resource Planning* (ERP) Odoo menyimpan data-data transaksi perusahaan. Namun, Odoo belum memiliki modul untuk mengelola data tersebut. Dibutuhkan sebuah modul untuk mengelola data menjadi informasi yang berguna.

Berdasarkan permasalahan di atas, maka dirancang modul *data mining Market Basket Analysis*. Modul ini menggunakan algoritma *FP-Growth* dengan memanfaatkan data transaksi penjualan.

Pengujian modul ini menggunakan data dari Supermarket X. Hasil akhir modul ini merupakan hasil dari proses *data mining* dalam bentuk *association rule*. *Rule* ditampilkan dalam bentuk narasi dan grafik, sehingga lebih mudah dipahami..

Kata Kunci: Odoo*, FP-Growth, Data Mining, Market Basket Analysis*.

ABSTRACT

Odoo Enterprise Resource Planning (ERP) system storing company's transaction data. However, Odoo doesn't have a module for managing data. It takes a module for managing data into useful information.

Based on the above problems, a module for data mining Market Basket Analysis is being designed. This module uses FP-Growth algorithm by utilizing the sales transaction data.

For the testing, this module using data from X Supermarket. The final result of this module is an association rule from data mining process. The rule is shown in the form of narrative and graphics making them easier to understand.

Keywords: Odoo*, FP-Growth, Data Mining, Market Basket Analysis*.

1. PENDAHULUAN

Sistem Informasi memiliki peranan penting dalam meningkatkan daya saing bisnis di berbagai bidang. Sistem *Enterprise Resource Planning* (ERP) menyediakan *platform* yang ideal untuk menjawab kebutuhan tersebut, namun resiko dan biaya untuk implementasi sistem ERP cukup tinggi. Resiko dan biaya tersebut dapat diminimalisasikan dengan penggunaan Odoo.

Odoo merupakan sistem ERP yang dibangun secara *open source*, oleh karena itu Odoo mendukung pemanfaatan kembali *library* yang telah dibuat sebelumnya. Kualitas yang dimiliki Odoo juga sangat baik karena banyaknya orang yang terlibat dalam pengembangannya. Sebanyak lebih dari 1500 pengembang tergabung dalam komunitas Odoo dan telah menghasilkan lebih dari 4500 modul untuk menjawab kebutuhan bisnis.

Dari sekian banyak modul yang dimiliki Odoo, masih belum ada modul yang dapat menjawab kebutuhan akan *Business Intelligence* yang menggunakan *data mining*. Oleh karena itu, penambahan fitur *Business Intelligence* yang menggunakan *data mining* melengkapi kemampuan Odoo untuk meningkatkan daya saing perusahaan.

Salah satu fungsi dari *data mining* yang digunakan untuk menganalisa penjualan adalah fungsi asosiasi atau yang sering disebut *Market Basket Analysis* (MBA). MBA menggunakan algoritma untuk menemukan produk-produk yang dibeli secara bersamaan. Pada skripsi ini, algoritma yang digunakan adalah *FP-Growth.* Pemilihan algoritma *FP-Growth* didasari oleh stabilitas dan kecepatan yang dimiliki oleh algoritma ini. Stabilitas yang dimaksudkan adalah algoritma ini ketika diujikan pada *low minimum support* dan *high minimum support* selalu memiliki performa yang baik.

Supermarket X adalah sebuah supermarket yang berada di daerah Nginden, Surabaya. Menjamurnya *minimarket* di Surabaya, termasuk di daerah Nginden, menjadikan persaingan antar supermarket semakin ketat. Supermarket X dengan omzet penjualan rata-rata Rp 8.503.422 perhari, membutuhkan suatu strategi yang tepat agar dapat terus bersaing. Oleh karena itu, Supermarket X cocok untuk dijadikan studi kasus dalam skripsi ini.

2. TINJAUAN PUSTAKA

2.1 Sistem Odoo

Odoo merupakan kumpulan dari aplikasi bisnis yang bersifat *open source*. Odoo dikembangkan oleh Odoo S.A. Odoo dibangun secara *open source*, oleh karena itu Odoo mendukung pemanfaatan kembali *library* yang telah dibuat sebelumnya. Selain itu, kualitas yang dimiliki Odoo juga sangat baik karena banyaknya orang yang terlibat dalam pengembangannya. Sebanyak lebih dari 1500 pengembang tergabung dalam komunitas Odoo dan telah menghasilkan lebih dari 4500 modul untuk menjawab kebutuhan bisnis pengguna. Pengguna Odoo berjumlah lebih dari 2.000.000 dan tersebar di seluruh dunia.[4]

Sistem Odoo terdiri dari tiga komponen utama, yaitu PostgreSQL, aplikasi server Odoo, dan web server [5]. Database PostgreSQL yang bersifat *open source* menampung semua data dan konfigurasi Odoo[6]. Aplikasi server Odoo berisi modul-modul yang menyusun Odoo. Web server merupakan aplikasi terpisah yang membuat user dapat mengakses aplikasi server Odoo secara langsung.

Fitur bisnis pada Odoo diorganisasi dalam bentuk modul-modul. Odoo menyediakan modul-modul dasar yang merupakan fungsi bisnis secara umum yang biasa digunakan

dalam perusahaan. Modul dasar tersebut dikelompokkan dalam 6 kelompok aplikasi:

- Aplikasi *front-end*
- Aplikasi *sales management*
- Aplikasi *business operations*
- Aplikasi *marketing*
- Aplikasi *human resource*

2.2 *Data Mining*

Data mining memiliki hakikat (*notion*) sebagi disiplin ilmu yang tujuan utamanya adalah untuk menemukan, menggali, atau menambang pengetahuan dari data atau informasi yang dimiliki [7]. Dalam rangka menemukan, menggali, atau menambang pengetahuan tersebut, data mining memiliki beberapa fungsi. Enam fungsi dalam data mining, yaitu fungsi deskripsi, fungsi estimasi, fungsi prediksi, fungsi klasifikasi, fungsi pengelompokan, dan fungsi asosiasi [3].

Data pada *database* operasional berasal dari banyak sumber data, sehingga sangat rentan terhadap data-data yang tidak konsisten seperti *field* yang kosong dan data yang tidak valid. Kualitas data yang rendah akan menyebabkan hasil *mining* yang kurang baik pula. Oleh karena itu, data perlu dipersiapkan terlebih dahulu melalui tahap *preprocessing* sebelum memasuki tahap *mining* [1].

Ada beberapa teknik *preprocessing* seperti *data cleaning* dan *data reduction*. *Data cleaning* digunakan untuk menghapus *noise* dan membenarkan data yang tidak konsisten. *Data reduction* adalah proses untuk mengurangi representasi *volume* data namun tidak mengurangi kualitas hasil analisis. Salah satu bagian dari *data reduction* adalah *data discretization* yang memfokuskan pada data *numeric* [1].

2.3 *Market Basket Analysis*

Market Basket Analysis (MBA) adalah salah satu metode dalam data mining yang fokusnya pada identifikasi produk-produk yang dibeli secara bersamaan dalam satu transaksi. Pada algoritma MBA terdapat istilah *support* dan *confidence*. *Support* adalah prosentase dari semua transaksi yang mengandung *itemset* yang dipilih. *Confidence* adalah prosentase dari semua transaksi yang mengandung *leading item* dan juga *depending item*. *Leading item* adalah produk yang dijadikan acuan.

Output dari MBA adalah sebuah rangkaian *rules* yang mengindikasikan produk-produk yang dibeli secara bersamaan. Algoritma ini termasuk dalam *The Top Ten Algorithms in Data Mining* [9]. MBA memiliki beberapa algoritma untuk menghasilkan *association rules,* misalnya Apriori dan *Find Pattern Growth (FP-Growth)*.

2.4 *Find Pattern Tree*

FP-Tree adalah sebuah representasi terkompresi dari data *input*. Setiap data transaksi dibaca, lalu dilakukan *mapping* ke dalam sebuah *path* pada *FP-Tree.* Transaksi-transaksi yang memiliki *item* yang sama akan menyebabkan *path* yang tumpang tindih. Semakin banyak *path* yang tumpang tindih, semakin terkompresi struktur *FP-Tree* yang terbentuk [8].

Tiap *node* pada *tree* menunjukkan nama *item*, *support counter* yang menunjukkan jumlah lintasan transaksi yang melalui *node* tersebut, dan *pointer* penghubung yang menghubungkan *node-node* dengan *item* yang sama antar *path.*

Berikut adalah algoritma untuk membangun *FP-Tree* [2].

Algoritma 1 (Pembangunan *FP-Tree*)

Input: Sebuah database transaksi DB dan *minimum support* yang dikehendaki

Output: *FP-Tree, frequent-pattern tree* dari DB

Metode: *FP-Tree* dibangun dengan cara:

1. *Scan* database transaksi DB sebanyak satu kali. Dapatkan *F*, *frequent items sets*, dan *support* dari tiap *frequent item*. Urutkan *support* dari *F* secara *descending*, dan masukkan ke dalam *FList,* daftar urutan dari *frequent items.*
2. Buat *root* dari *FP-Tree* dan beri label sebagai "null". Bagi setiap transaksi *Trans* dalam DB, diperlakukan sebagai berikut:

- Tiap *frequent item* dalam *Trans* diurutkan berdasarkan urutan pada *FList*. Daftar *frequent item* dalam *Trans* yang telah diurutkan, didefinisikan sebagai [p|P], dimana p adalah *element* pertama dan P adalah daftar yang tersisa. Panggil fungsi *insert_tree*([p|P], T]).
- Fungsi *insert_tree*([p|P], T]) berjalan sebagai berikut. Jika T memiliki anak N dimana N.item-name = p.item-name, lalu tambahkan *count* dari N sebanyak satu; *else* buat sebuah *node* baru N dengan *count* sebesar satu, lalu hubungkan *parent link* dari N ke T, dan hubungkan *node-link* N ke *node-node* lain yang memiliki *item-name* yang sama via *node-link structure*. Jika P masih belum kosong, panggil *insert_tree*([p|P], T]) secara rekursif.

2.5 *Find Pattern Growth* (*FP-Growth*)

Algoritma *Find Pattern Growth (FP-Growth)* adalah sebuah algoritma alternatif yang menggunakan pendekatan yang berbeda secara radikal untuk menemukan *frequent itemsets* [8]. Algoritma ini tidak menggunakan paradigma *generate-and-test* yang digunakan pada Apriori. *FP-Growth* namun menggunakan struktur data khusus yaitu *Find Pattern Tree (FP-Tree)*. *Frequent itemsets* diekstrak secara langsung dari *FP-Tree*. Oleh karena itu, algoritma *FP-Growth* lebih cepat daripada algoritma Apriori.

Zheng, Kohavi, dan Mason melakukan perbandingan terhadap performa dari empat algoritma *Market Basket Analysis*: Apriori, *FP-Growth,* Charm, dan Closet. Performa diukur dari seberapa cepat sebuah algoritma menghasilkan *frequent itemsets* dari empat *dataset.* Tabel 1 menunjukkan hasil percobaan yang dilakukan dengan *high minimum supports* dan *low minimum supports.* Dari setiap percobaan algoritma *FP-Growth* tidak pernah memiliki performa yang paling rendah sehingga dapat disimpulkan, algoritma *FP-Growth* merupakan algoritma yang stabil dalam berbagai situasi [10].

Tabel 1. Ranking kemampuan algoritma [10]

(Ap: Apriori, FP: FP-Growth, Ch: Charm, Cl: Closet)

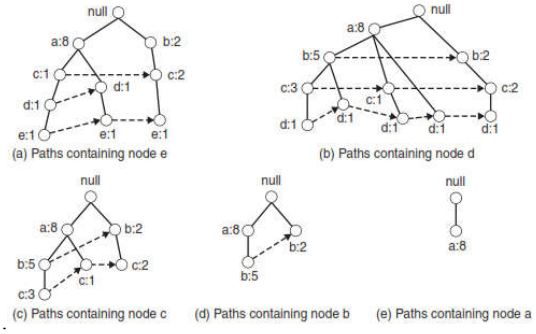
	High Min-Support	**Low Min-Support**
IBM-Artificial	Ap > FP > Ch > Cl	FP > Ch > Cl > Ap
BMS-POS	Ap > Cl > FP > Ch	Ch > FP > Ap > Cl
BMS-WebView-1	Ap > FP > Cl > Ch	Ch > FP > Ap > Cl
BMS-WebView-2	Ap > FP > Ch > Cl	Ch > FP > Ap > Cl

Langkah-langkah penting dalam algoritma FP-Growth yaitu:

1. Membentuk *conditional pattern base* atau *subproblem* untuk tiap *node* dalam *FP-Tree*

   Pada tahap pertama ini, *FP-Tree* yang telah lengkap sebelumnya dipecah menjadi beberapa *subproblem*. Jumlah

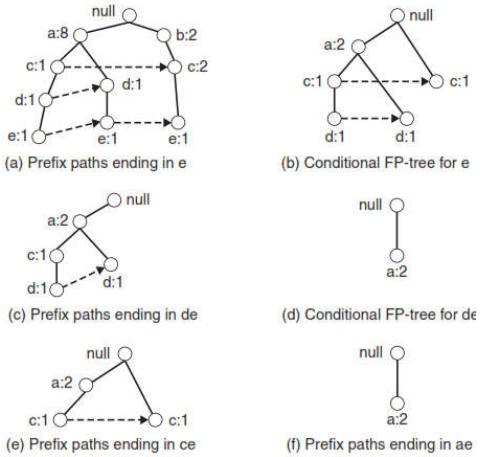
subproblem sama dengan jumlah *item*. Hasil dari tahap pertama ini dapat dilihat pada Gambar 1 [8].



Gambar 1. Conditional Pattern Base untuk Tiap Item [8]

2. Membentuk conditional FP-Tree untuk tiap *conditional pattern base* dan secara *recursive* melakukan *mining* pada *conditional FP-Tree*

   Dari setiap *conditional pattern base,* dibentuklah *conditional FP-Tree.* Misalnya pada Gambar 2 diambil contoh pencarian *frequent itemsets* untuk *path* yang berakhir dengan *node e*. Pertama harus dipastikan bahwa *itemset* {*e*} memenuhi *minimum support* atau *frequent.* Setelah itu dibentuk pula *subproblem* untuk mencari *frequent itemsets* yang berakhir pada *de*, *ce*, *be,* dan *ae*. Tiap *subproblem* dipecah menjadi *subproblem* yang lebih sederhana. Dengan menggabungkan solusi didapat dari semua *subproblem,* maka dapat diperoleh semua *frequent itemsets* yang berakhir pada *e* [8].



Gambar 2. Pengaplikasian Algoritma *FP-Growth* untuk Mencari *Frequent Itemsets* yang Berakhir di *Node e* [8]

3. Menghitung support untuk tiap *frequent pattern*
   Pada tahap terakhir, dilakukan perhitungan untuk tiap *frequent pattern* yang telah dihasilkan. Hasil inilah yang merupakan *output* dari algoritma *FP-Growth.*

2.6 Algoritma *FP-Growth*

FP-Growth merupakan salah satu algoritma untuk menghasilkan *association rule.* Algoritma ini melakukan *mining frequent pattern* menggunakan *FP-Tree* yang telah dibangun sebelumnya [2].

Algoritma 2 (*FP-Growth: Mining frequent pattern* menggunakan *FP-Tree* dengan *pattern fragment growth*).

Input: Sebuah database DB yang direpresentasikan oleh *FP-*Tree yang telah dibangun berdasarkan Algoritma 1 dan *minimum support* yang dikehendaki

Output: Kumpulan lengkap dari *frequent pattern*

Metode: Memanggil fungsi *FP-Growth(Tree, null)*.

Procedure FP-Growth(*Tree, α*)
{if *Tree* mengandung *path* yang memiliki *single prefix*
then {
P = bagian dari *Tree* yang memiliki *single prefix;*
Q = bagian dari *Tree* yang *multipath,* mengganti *node* paling atas dengan *null root;*
for tiap kombinasi (dinotasikan sebagai *β*) dari *nodes* dalam *path P do*
generate pattern β ∪ *α* dengan *support* = *minimum support* dari *nodes* dalam *β;*
freq_pattern_set(*P*) = hasil dari *generate pattern β*; }
else Q = *Tree;*
for tiap *item* a_i dalam *Q do* {
generate pattern $\beta = a_i \cup \alpha$ dengan *support* = a_i.*support;*
construct conditional pattern-base dari β dan *conditional FP-Tree* dari *β* $Tree_\beta$;
if $Tree_\beta$ <> *empty*
then panggil *FP-growth*($Tree_\beta$, *β*);
freq_pattern_set(*Q*) = hasil dari *generate pattern β*; }
return (*freq_pattern_set*(*P*) ∪ *freq_pattern_set*(*Q*)∪ (*freq_pattern_set*(*P*) x *freq_pattern_set*(*Q*))}

3. ANALISIS DAN DESAIN

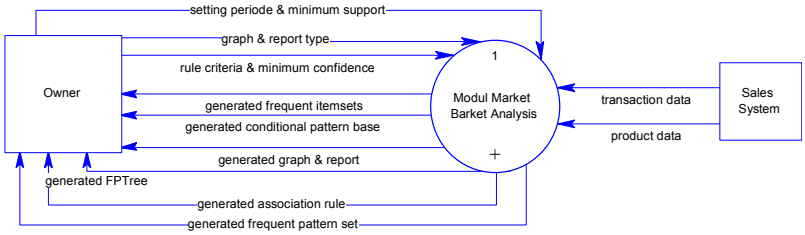
3.1 Analisis Perusahaan

Supermarket X memiliki lebih dari 16.000 macam barang dengan omzet penjualan rata-rata Rp 8.503.422 perhari. Supermarket X telah memiliki sistem yang terkomputerisasi sejak tahun 2006, sehingga semua data transasksi telah disimpan ke dalam *database*. Data yang sangat banyak tersebut tidak banyak dimanfaatkan untuk menghasilkan informasi yang lebih bermakna. Promosi yang dilakukan terhadap barang-barang yang dijual seringkali hanya mengikuti promosi dari supplier. *Bundling* produk juga hanya berdasarkan perkiraan. Selain itu, pembelian barang juga berdasarkan keadaan sekarang dan perkiraan. *Owner* belum memanfaatkan data-data transaksi sebelumnya untuk proses pengambilan keputusan. Oleh karena itu, Supermarket X membutuhkan aplikasi yang dapat mengolah data transaksi tersebut menjadi informasi yang berguna bagi *top-level management* dalam proses pengambilan keputusan.

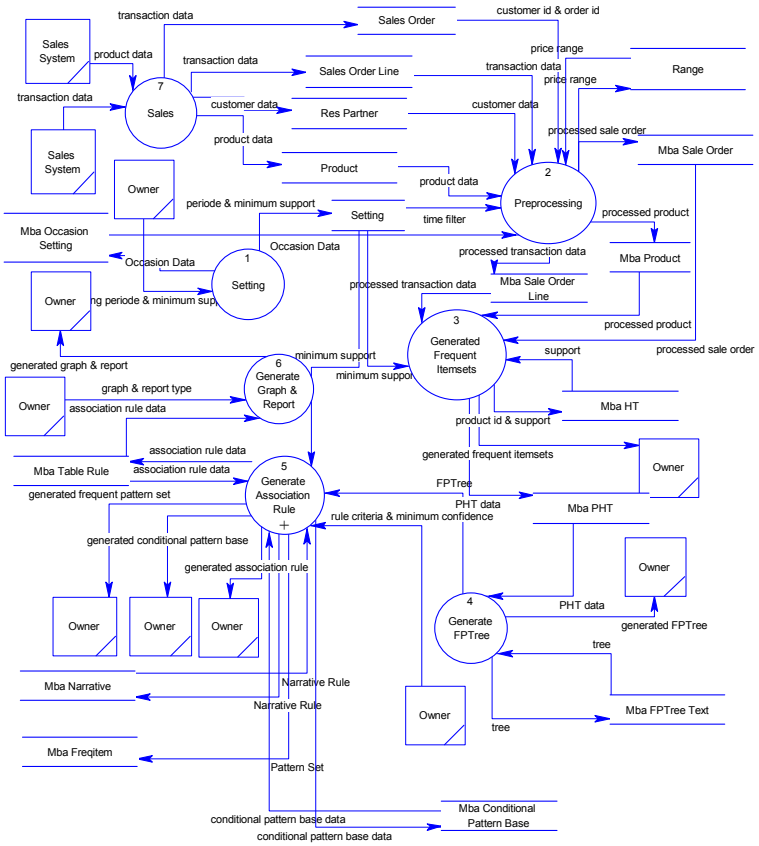
3.2 Analisis Kebutuhan

Dari permasalahan-permasalahan yang ada di atas, dapat disimpulkan bahwa *Owner* Supermarket X membutuhkan suatu sistem yang berbasis komputer untuk membantu dalam pengambilan keputusan. Kriteria sistem sebagai berikut:

- Aplikasi data mining yang dapat menghasilkan informasi berupa tingkat asosiasi antardata barang. Sistem yang dibutuhkan mempunyai konsep multidimensi yang menunjukkan hubungan yang ada. Dimensi yang digunakan adalah dimensi barang, *customer, supplier* dan waktu.
- Sistem yang dibutuhkan dapat memberikan informasi mengenai transaksi barang kepada *Owner* Supermarket X.

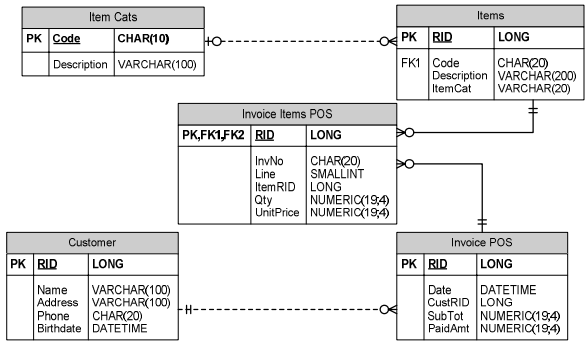


Gambar 3. *Context Diagram*



Gambar 4. Data Flow Diagram Level 0

Sumber data yang digunakan dalam Supermarket X dapat dilihat pada Gambar 5. Terdapat lima tabel yang berupa menyimpan data kategori barang, data barang, data pelanggan, data penjualan dan data detail pejualan.



Gambar 5. ERD Sistem Penjualan Supermarket X

3.3 Desain *Sistem*

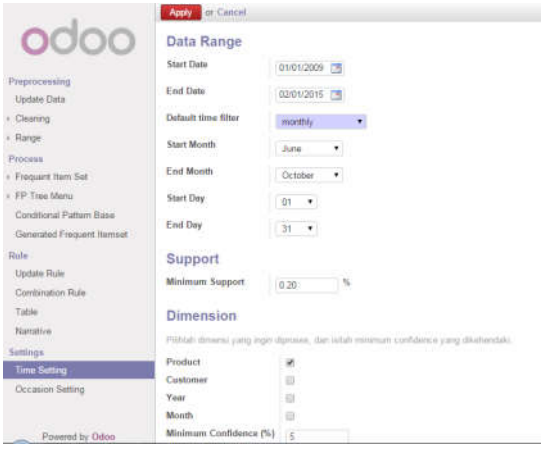
3.3.1 Data Flow Diagram (DFD)

Perancangan modul dimulai dari pembuatan desain keseluruhan sistem meggunakan DFD. Aliran data pada modul secara keseluruhan digambarkan oleh *Context Diagram* pada Gambar 3. Penjelasan lebih detail dapat dilihat pada subbab berikutnya, yaitu DFD Level 0 pada Gambar 4.

4. PENGUJIAN APLIKASI

Pengujian ini dilakukan untuk membuktikan kebenaran aplikasi yang telah dibuat. Tampilan aplikasi secara keseluruhan dapat dilihat pada Gambar 6. Pengujian dilakukan dengan membandingkan hasil pencarian frequent itemset pada aplikasi dengan hasil dalam buku[1]. Data tidak real dibuat mirip dengan yang ada di dalam buku untuk pengujian ini.

Sebagai catatan, ada beberapa perbedaan antara pengujian pada aplikasi dengan data yang terdapat dalam buku, yaitu: contoh yang terdapat pada buku buku menampilkan id *item* pada HT, *tree*, *Conditional Pattern Base*, dan *Frequent Pattern Generated*, sedangkan pada program yang ditampilkan merupakan id HT.



Gambar 6. Tampilan Aplikasi

Pada Tabel 2 ditunjukkan *mapping* antara id HT dan id *item*.

- Program dirancang untuk multi-dimensi data mining, sedangkan yang terdapat di paper hanya dimensi item. Oleh karena itu, maka ditambahkan dimensi waktu yang ditentukan oleh penulis.
- Pada paper, tiap item hanya memiliki 1 level sedangkan program dirancang untuk melakukan proses ke dalam 5 level yaitu level 1 dimensi barang, level 2 department, level 3 category, level 4 range, level 5 item. Oleh karena itu, data dari paper akan dimodifikasi oleh penulis agar memenuhi ke-5 level tersebut.

Tabel 2. *Mapping* antara id HT dan id *item*

Id HT	Id *Item*
1	3
2	4
3	2
4	5
5	1

Pada Gambar 7 dapat dilihat data detail penjualan dari buku. Terdapat sembilan transaksi penjualan berupa T100 sampai T900. Dari sembilan transaksi tersebut dapat dilihat bahwa terdapat lima macam produk, antara lain: I1, I2, I3, I4, dan I5.

TID	*List of item_IDs*
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

Gambar 7. Data Detail Penjualan dari Buku [1]

Berdasarkan data contoh yang berasal dari buku, dimasukkan data produk ke dalam aplikasi. Pada aplikasi sudah ada produk lain, yaitu produk *Service,* sehingga produk I1 yang dimasukkan memiliki id 2, produk I2 memiliki id 3, dan seterusnya hingga produk I5. Data produk pada aplikasi dapat dilihat pada Gambar 8.

	id [PK] serial	ean13 character varying(13)	create_date timestamp without time zone	default_code character varying	name_template character varying
1	1		2015-11-25 03:43:19.159		Service
2	2		2015-11-25 03:49:18.959		I1
3	3		2015-11-25 03:49:38.546		I2
4	4		2015-11-25 03:50:42.103		I3
5	5		2015-11-25 03:50:48.071		I4
6	6		2015-11-25 03:51:12.978		I5

Gambar 8. Data Produk pada Aplikasi

Setelah memasukkan data produk, kemudian data penjualan juga dimasukkan sesuai dengan data contoh pada buku. Pada buku tidak dicantumkan tanggal terjadinya transaksi, namun pada program diwajibkan mengisi tanggal transaksi, sehingga diisi dengan tanggal 25 November 2015. Data penjualan pada aplikasi dapat dilihat pada Gambar 9.

	id integer	name character varying
1	1	SO001
2	2	SO002
3	9	SO009
4	3	SO003
5	4	SO004
6	5	SO005
7	6	SO006
8	7	SO007
9	8	SO008

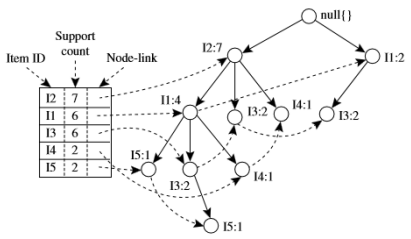
Gambar 9. Data Penjualan pada Aplikasi

Detail dari sembilan transaksi penjualan yang telah dimasukkan ke dalam aplikasi dapat dilihat pada Gambar 10.

	order_id integer	product_name text
1	1	I2
2	1	I5
3	1	I1
4	2	I4
5	2	I2
6	3	I3
7	3	I2
8	4	I4
9	4	I1
10	4	I2
11	5	I1
12	5	I3
13	6	I2
14	6	I3
15	7	I3
16	7	I1
17	8	I2
18	8	I1
19	8	I3
20	8	I5
21	9	I3
22	9	I2
23	9	I1

Gambar 10. Data Detail Transaksi pada Aplikasi

Pada Gambar 11 dapat dilihat isi dari *Heaader Tablei* dan *tree* yang terbentuk dari contoh yang beasal dari buku. Ditampilkan bahwa terdapat lima *item* yang berada di dalam *Header Table* dengan urutan seusai dengan *support count* mulai dari yang terbesar hingga yang terkecil. Kemudian berdasarkan data transaksi dan data dari tabel ini, dibangunlah *tree*.



Gambar 11. *Header Table* dan *Tree* dari Buku [1]

Header Table yang dihasilkan pada aplikasi dapat dilihat pada Gambar 12. Namun perlu dilakukan *mapping* untuk mengetahui persamaan hasil antara aplikasi dengan buku. Mapping dilakukan berdasarkan Tabel 2

ID	Id Item	Type	Count	Order
3	3	product	7	1
5	2	product	6	2
1	4	product	6	3
4	6	product	2	4
2	5	product	2	5

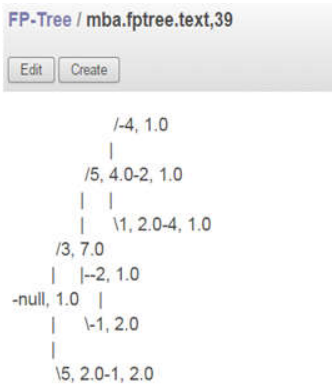
Gambar 12. *Header Table* pada Aplikasi

Setelah dilakukan *mapping,* maka dihasilkan *Header Table* seperti pada Tabel 3. Hasil ini menunjukkan bahwa *Header Table* yang dihasilkan aplikasi setelah dilakukan *mapping* telah seusai dengan hasil *Header Table* yang ditercantum pada buku.

Tabel 3. *Header Table* yang Telah Disesuaikan

Id	Id Item	Type	Count	Order
3	I2	product	7	1
5	I1	product	6	2
1	I3	product	6	3
4	I5	product	2	4
2	I4	Product	2	5

Pada Gambar 13 dapat dilihat *tree* yang dihasilkan oleh aplikasi. *Tree* yang dihasilkan sudah sesuai dengan yang berada pada buku, yaitu yang ditunjukkan pada Gambar 11.



Gambar 13. *Tree* pada Aplikasi

Pada Gambar 14 ditunjuukan *Conditional Pattern Base, Conditional FP-Tree,* dan *Frequent Pattern Set* yang dicantumkan dalam buku.

Item	*Conditional Pattern Base*	*Conditional FP-tree*	*Frequent Patterns Generated*
I5	{{I2, I1: 1}, {I2, I1, I3: 1}}	⟨I2: 2, I1: 2⟩	{I2, I5: 2}, {I1, I5: 2}, {I2, I1, I5: 2}
I4	{{I2, I1: 1}, {I2: 1}}	⟨I2: 2⟩	{I2, I4: 2}
I3	{{I2, I1: 2}, {I2: 2}, {I1: 2}}	⟨I2: 4, I1: 2⟩, ⟨I1: 2⟩	{I2, I3: 4}, {I1, I3: 4}, {I2, I1, I3: 2}
I1	{{I2: 4}}	⟨I2: 4⟩	{I2, I1: 4}

Gambar 14. *Conditional Pattern Base, Conditional FP-Tree, Frequent Pattern* dari Buku [1]

Pada Gambar 15 ditunjukkan *Conditional Pattern Base* yang dihasilkan oleh aplikasi. Angka pada kolom *Item* dan *Conditional Pattern Base* menunjukkan id HT.

Item	Conditional Pattern Base	Support
5	3,	4
2	3, 5,	1
2	3,	1
1	3, 5,	2
1	3,	2
1	5,	2
4	3, 5,	1
4	3, 5, 1,	1

Gambar 15. Conditional Pattern Base pada Aplikasi

Pada Tabel 4 dapat dilihat hasil *Conditional Pattern Base* yang telah disesuaikan dengan cara mengubah id HT yang terletak pada kolom *Item* dengan id *item*. Hasil ini menunjukkan bahwa *Conditional Pattern Base* yang dihasilkan aplikasi telah sesuai dengan yang dicantumkan dalam buku, seperti yang dapat dilihat pada Gambar 14.

Tabel 4. Conditional Pattern Base yang Telah Disesuaikan

Id *Item*	*Conditional Pattern Base*	*Support*
I1	I2	4
I4	I2, I1	1
I4	I2	1
I3	I2, I1	2
I3	I2	2
I3	I1	2
I5	I2, I1	1
I5	I2, I1, I3	1

Gambar 16 dapat dilihat *Frequent Pattern Generated* yang dihasilkan pada aplikasi. Data pada kolom *Set* menampilkan id HT, pada kolom *Support* merupakan jumlah *count* tiap *frequent pattern set*. Karena kolom *set* menampilkan id HT, maka dibutuhkan penyesuaian terlebih dahulu sebelum dibandingkan dengan hasil yang ada pada buku

Set	Support
3,	7
5,	6
3,5,	4
2,	2
2,5,	1
2,3,5,	1
2,3,	2
1,	6
1,3,	4
1,5,	4
1,3,5,	2
4,	2
1,4,	1
1,4,5,	1
1,3,4,5,	1
1,3,4,	1
4,5,	2
3,4,5,	2
3,4,	2

Gambar 16. Frequent Pattern Generated pada Aplikasi

Pada Tabel 5 dapat dilihat hasil *Frequent Pattern Generated* yang telah diseusaikan. Pada Gambar 14 dapat dilihat bahwa *Frequent Pattern Generated* yang dihasilkan aplikasi telah sesuai dengan yang dicantumkan dalam buku.

Tabel 5. Frequent Pattern Generated yang Telah Disesuaikan

Set	*Support*
I2	7
I1	6
I2, I1	4
I4	2
I4, I1	1
I4, I2, I1	1
I4, I2	2
I3	6
I3, I2	4
I3, I1	4
I3, I2, I1	2
I5	2
I3, I5	1
I3, I5, I1	1
I3, I2, I5, I1	1
I3, I2, I5	1
I5, I1	2
I2, I5, I1	2
I2, I5	2

5. KESIMPULAN

Dari hasil perancangan dan pembuatan aplikasi, dapat diambil kesimpulan antara lain:

1. Pembuatan aplikasi mampu melengkapi fitur dari Odoo.
2. Algoritma *FP-Growth* yang digunakan pada aplikasi dapat menghasilkan *multidimensional association rules.*
3. Semakin kecil *minimum support* yang ditentukan, semakin banyak *frequent itemset* yang dihasilkan, sehingga proses *mining* semakin lama.
4. Hasil dari kuesioner menunjukkan tampilan aplikasi 100% baik, fungi kustomisasi grafik 75% baik, fungsi kustomisasi hubungan antardata 100% baik, kemudahan penggunaan aplikasi 50% baik, petunjuk yang diberikan 75% baik, dan keseluruhan aplikasi 100% baik. Hal ini menunjukkan bahwa aplikasi yang dikembangkan dapat diaplikasikan dengan baik.
5. Semakin kecil *minimum support* yang ditentukan, semakin banyak *frequent itemset* yang dihasilkan, sehingga proses *mining* semakin lama.

6. DAFTAR PUSTAKA

[1] Han, J., Kamber, M., & Pei, J. (2012). *Data Mining: Concepts and Techniques (3rd ed.).* San Fransisco: Morgan Kaufman.

[2] Han, J., Pei, J., Yin, Y., & Mao, R. (2004). Mining Frequent Patterns without Candidate Generation: A Frequent-Pattern Tree Approach. *Data Mining and Knowledge Discovery 8*, 53-87.

[3] Larose, D. T. (2005). *Discovering Knowledge in Data: An Introduction to Data Mining.* Hoboken: Wiley-Interscience, Jogn Wiley & Sons, Inc.

[4] Moss, G. (2013). *Working with OpenERP.* Brimingham: Packt Publishing Ltd

[5] Reis, D. (2015). *Odoo Development Essentials.* Brimingham: Packt Publishing.

[6] Riggs, S., & Krossing, H. (2010). *PostgreSQL 9 Administration Cookbook.* Brimingham: Packt Publishing Ltd.

[7] Susanto, S., & Suraydi, D. (2010). *Pengantar Data Mining Menggali Pengetahuan dan Bongkahan Data.* Yogyakarta: Andi.

[8] Tan, P. T., Steinbach, M., & Kumar, V. (2005). *Introduction to Data Mining.* New York: Pearson.

[9] Wu, X. D., & Kumar, V. (2009). *The Top Ten Algorithms in Data Mining.* Boca Raton: Chapman and Hall/ CRC.

[10] Zheng, Z., Kohavi, R., & Mason, L. (2001). Real World Performance of Association Rule Algorithms. *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining 7*, 401-405.