

Analytical Processing pada Graph Dataset dengan Metode Graph OLAP Framework

Analytical Processing on Graph Dataset with Graph OLAP framework Methodology

Ilham Kamil Noviandi¹, Kemas Rahmat S W., ST, M.Eng², Gde Agung A W., S.Kom, M.T³

Departemen Teknik Informatika, Universitas Telkom

Jalan Telekomunikasi No.1, Dayeuhkolot Bandung 42057 Indonesia

ilhamkamiln@gmail.com, bagindokemas@telkomuniversity.ac.id, ary.wisudiawan@gmail.com

Abstrak

Kehandalan penerapan *Online Analytical Processing* (OLAP) pada database relasional dalam menangani volume data yang besar. operasi *data cube* dan operasi-operasi pendukung khas OLAP seperti *roll-up*, *drill-down*, *slice and dice* memberikan *multidimensional* dan *multi-level view* yang memudahkan analisis kompleks pada *bussiness intelligence*, memunculkan banyaknya permintaan penerapan OLAP pada *graph database*.

Graph database itu sendiri telah menjadi tren saat ini karena kemampuannya dalam memproses data yang bersifat struktural dan semi struktural dibandingkan dengan *relational database* yang menggunakan *recursive join* sehingga menyebabkan komputasi semakin kompleks.

Banyaknya permintaan dan kelebihan dari *graph database* dalam memodelkan dataset yang berisi informasi terstruktur, maka pada penelitian tugas akhir ini akan menerapkan *graph OLAP framework*. *Graph OLAP framework* akan merepresentasikan *graph database* dalam bentuk *snapshots graph* dan beberapa operasi-operasi khas OLAP yang memberikan *multidimensional* dan *multi-level view*.

Kata Kunci: *Online Analytical Processing, data, roll-up, drill down, slice and dice, multidimensional, multi-level view, bussiness intelligence, graph database, graph OLAP framework, snapshots*

Abstract

Reliability of *Online Analytical Processing* (OLAP) on *relational database* in handling a large volume of data. Data cube operations and operation typical of OLAP such as, *roll-up*, *drill-down*, *slice and dice* give a *multidimensional* and *multi-level view* which allows complex analysis in *bussiness intelligence*, appeared many demands that using OLAP on *graph database*.

Graph Database itself has become the current trend because its ability to process data that is both structural and semi-structural compared to relational database that uses recursive join which is causing increasingly complex computing.

Many demands and an excess of *graph database* in modeling dataset that contain structure information. So in this final project, it will apply *graph OLAP framework*. *Graph OLAP framework* will represent *graph database* in the form of *snapshots graph* and some distinctive operations of OLAP which gives *multidimensional* and *multi-level view*

Keyword : *Online Analytical Processing, data, roll-up, drill down, slice and dice, multidimensional, multi-level view, bussiness intelligence, graph database, graph OLAP framework, snapshots*

1. Pendahuluan

Beberapa tahun ini *Graph Database* merupakan salah satu trend bisnis yang digunakan untuk bisnis berskala besar[8]. Dikarenakan graph database tidak hanya sangat berguna dalam memahami dataset yang memiliki keragaman data yang luas dan juga untuk data yang memiliki ukuran dan nilai yang sudah pasti, graph database merupakan cara yang terbaik untuk merepresentasikan dan memproses data yang berhubungan [8].

Sudah terbukti bahwa database relasional memiliki response time yang sangat lama dibandingkan dengan graph database apabila dimensi pencariannya semakin dalam [8]. Jaringan dari graph database dapat dimodelkan sebagai graph besar dengan simpul mewakili entitas dan sisi yang menggambarkan hubungan antara entitas[3]. Sehingga response time dari graph database lebih cepat dibandingkan dengan database relasional yang menggunakan

recursive joins yang menyebabkan komputasi semakin kompleks [8].

Kehandalan penerapan *Online Analytical Processing* (OLAP) pada database relasional dalam menangani volume data yang besar. operasi *data cube* dan operasi-operasi pendukung khas OLAP seperti *roll-up*, *drill-down*, *slice* and *dice* memberikan *multidimensional* dan *multi-level view* yang memudahkan analisis kompleks pada *bussiness intelligence*, memunculkan banyaknya permintaan penerapan OLAP pada *graph database*[2].

Banyaknya permintaan dan kelebihan dari *graph* dalam memodelkan dataset yang berisi informasi terstruktur, maka pada penelitian tugas akhir ini akan menerapkan *graph OLAP framework*. *Graph OLAP framework* akan merepresentasikan *graph database* dalam bentuk *snapshots graph* dan beberapa operasi pada OLAP klasik yang memberikan *multidimensional* dan *multi-level view*. Oleh karena itu, judul yang disusun penulis adalah “Analytical Processing pada Graph Dataset dengan Metode Graph OLAP Framework”.

1.2 Perumusan Masalah

Permasalahan yang akan diselesaikan dalam Tugas Akhir ini adalah :

1. Bagaimana mengagregasikan *graph*?
2. Bagaimana menerapkan operasi *roll-up* dan *drill-down* pada dataset sesuai dengan metode Graph OLAP Framework?
3. Bagaimana menerapkan *localization* pada dataset?
4. Bagaimana *me-ranking* node berdasarkan frekuensi kemunculan?
5. Bagaimana menerapkan operasi *query* pencarian barang?

1.3 Tujuan

Tujuan yang ingin dicapai dalam penelitian Tugas Akhir ini adalah :

1. Dapat mengagregasikan *graph*.
2. Dapat menerapkan operasi *roll-up* dan *drill-down* pada dataset.
3. Dapat menerapkan *localization* pada dataset.
4. Dapat *me-ranking* node berdasarkan frekuensi kemunculan.
5. Menerapkan operasi *query* pencarian barang.

2. Tinjauan Pustaka

2.1 Teori Graph

2.1.1 Sejarah singkat teori graph

Teori *Graph* mulai dikenal saat matematikawan kebangsaan Swiss bernama Leonhard Euler, yang berhasil mengungkapkan Misteri Jembatan Koningsberg tahun 1736. Di kota Koningsberg mengalir sungai Pregel, di sungai mengalir pulau dan diantaranya terdapat jembatan yang menghubungkan, jumlah jembatan tersebut sebanyak 7 buah [6].

2.1.2 Dasar teori graph

Sebuah *graph linier* (atau secara sederhana disebut *graph*) $G = (V, E)$ terdiri atas sekumpulan objek $V = \{v_1, v_2, \dots\}$ yang disebut himpunan titik, dan sebuah himpunan lain $E = \{e_1, e_2, \dots\}$ yang merupakan himpunan sisi sedemikian hingga tiap sisi ek dikaitkan dengan suatu pasangan tak terurut (v_i, v_j) . Titik v_i, v_j yang berkaitan dengan ek disebut titik-titik ujung sisi ek. Cara merepresentasikan sebuah *graph* yang paling umum adalah berupa diagram. Dalam diagram tersebut, titik-titik dinyatakan sebagai noktah dan tiap sisi dinyatakan sebagai segmen garis yang menghubungkan tiap dua titik [6].

Dalam sebuah *graph*, seperti terlihat pada contoh di atas, dimungkinkan adanya suatu sisi yang dikaitkan dengan pasangan (v_1, v_2) . Sisi yang dua titik ujungnya sama disebut loop. Dalam *graph* pada Gambar 2.1, e_1 merupakan sebuah loop. Dari contoh di atas juga perlu dicatat bahwa dalam sebuah *graph* dimungkinkan adanya lebih dari satu sisi yang dikaitkan dengan sepasang titik. Sebagai contoh, e_4 dan e_5 pada *graph* di atas dikaitkan dengan pasangan titik (v_1, v_3) . Pasangan sisi semacam ini disebut sisi-sisi paralel atau sejajar[6].

Sebuah *graph* yang tidak memiliki loop dan sisi paralel disebut *graph sederhana*. Dalam beberapa literatur teori *graph*, pembahasan hanya dibatasi pada *graph sederhana*, akan tetapi dalam banyak aplikasi teknik, penggunaan loop dan sisi paralel sangat diperlukan. Untuk membedakan antara *graph* yang memuat loop dan sisi paralel dengan *graph* yang tidak memuat kedua hal tersebut, sebagian penulis menggunakan istilah *graph sederhana* untuk *graph* yang

tidak memuat loop dan sisi paralel, serta graph umum untuk yang lainnya[6].

Dalam sebuah graph mungkin terdapat dua sisi atau lebih yang menghubungkan dua titik yang berlainan. Kedua sisi tersebut dinamakan sisi rangkap atau sisi ganda. Graph yang mengandung loop atau sisi rangkap dinamakan graph ganda[6].

2.2 Graph Database

Salah satu penerapan teori *graph* pada bidang teknologi informasi adalah *graph database*, yaitu struktur data yang menggunakan *graph* untuk merepresentasikan entitas satu objek sebagai *node* dan bagaimana entitas itu berhubungan dengan lingkungannya (antar objek), yang mana hubungan tersebut sebagai *edge*. Keunggulan *graph database* dibanding database lainnya adalah berupa kemampuan dalam memodelkan berbagai macam skenario pada dunia nyata, mulai dari sistem jalan raya, jaringan telepon, *World Wide Web*, serta yang belakangan ini menjadi tren pada *social networking*, *system recommendation* dan banyak lainnya. Dan juga kehandalannya dalam mengolah data-data yang bersifat tidak terstruktur, yang mana tidak terdapat pada *database relational* misalnya.

Model *graph* yang paling banyak digunakan [8] pada *graph database* adalah model *property graph*, yaitu struktur *graph* yang terdiri atas *node*, *edge*, dan *properties* untuk merepresentasikan dan menyimpan data.

Menurut Ian Robinson [8] sebuah sistem manajemen *graph database* merupakan sistem manajemen *database online* dengan metode CRUD (*Create*, *Read*, *Update* dan *Delete*) yang mengekspos sebuah model data *graph*. Oleh sebab itu, *graph database* biasanya dioptimalkan untuk performa transaksi dan dirancang dengan integritas transaksi dan ketersediaan operasional yang diperhatikan. Berikut ini adalah kekuatan dari *graph database* menurut Ian Robinson [8]:

1. Performa

Salah satu alasan kuat memilih *graph database* adalah peningkatan kinerja yang tipis ketika berurusan dengan data terhubung ketika dibandingkan antar *graph database* dengan *database relational*. Berlawanan dengan *database relational* dimana performa *query join* yang besar semakin memburuk dengan semakin besarnya dataset. Pada *graph database*, kinerjanya cenderung tetap relatif konstan bahkan ketika dataset semakin membesar. Hal ini karena *query-query* dilokalisasi ke bagian dari *graph*.

Akibatnya, waktu eksekusi untuk setiap *query* sebanding dengan ukuran dari bagian *graph* yang ditelusuri untuk memenuhi *query* tersebut, bukan dari ukuran keseluruhan *graph*.

2. Fleksibilitas

Graph secara alami bersifat aditif, yang berarti kita dapat menambahkan jenis hubungan baru (*relationship*) dan *subgraph* yang baru ke struktur yang ada tanpa mengganggu *query-query* dan fungsionalitas aplikasi yang ada.

3. Ketangkasan

Graph database modern memperlengkapi kita untuk melakukan pembangunan dan pemeliharaan sistem. Secara khusus, sifat skema bebas dari model data *graph* ditambah dengan sifat uji aplikasi antarmuka pemrograman *graph database* (API) dan bahasa *query* yang memberdayakan kita untuk mengembangkan aplikasi dengan cara yang terkendali.

2.3 Online Analytical Processing (OLAP)

Menurut Michal Bachman [2] tujuan utama dari *Online Transactional Processing* (OLTP) adalah eksekusi dari transaksional yang relatif simpel dengan skala besar namun efisien. Disini termasuk operasi *create*, *read*, *update*, dan *delete* (CRUD).

OLTP pada *relational database*, semua operasi CRUD biasanya terbatas hanya menggunakan *Structured Query Language* (SQL). *Read query* diimplementasikan untuk pencarian relasi (*tables*) dan menghasilkan relasi sementara dengan memproyeksikan *attributes* dan *tuples* yang dipilih. Ketika lebih dari satu relasi pada *read query*, biasanya operasi *join* seringkali digunakan. Adanya operasi ini merupakan salah satu karakteristik yang membedakan database relasional. Namun, terkadang operasi *join* tersebut seringkali menjadi penghambat kinerja[2].

OLTP pada *graph database*, tidak ada standar tunggal penggunaan *query language* di *graph database*. Operasi CRUD biasanya dilakukan menggunakan *query language* yang spesifik terhadap database (atau beberapa database), selain itu menggunakan RESTful atau *programming-language-specific* API. OLTP pada *graph databases*, *read query* diimplementasikan pada *property graph*. Hasil dari *read query* pada database biasanya adalah baik dalam satu set *vertices*, *edges*, atau *paths*[2].

Berbeda dengan OLTP, *online analytical processing* (OLAP) adalah sebuah *dynamic synthesis*, analisis, dan penggabungan dari *multidimensional data* yang bervolume besar. Sulit untuk menentukan batasan yang jelas antara operasi mana yang masih OLTP dan operasi mana yang dapat dikatakan OLAP, tetapi biasanya OLAP melibatkan *multidimensional view* pada data yang diagregasi untuk tujuan penganalisisan data. Sesuatu yang berupa proses pencarian sederhana di *graph database* (dapat dianggap sebagai OLTP) dapat menjadi operasi *multi-join* yang kompleks di database relasional (sehingga memenuhi kualifikasi sebagai OLAP). Di sisi lain, sebuah operasi SUM yang simpel pada sebuah *attribute* di semua *tuples* dalam satu relasi mungkin masih dianggap OLTP di database relasional, tetapi akan diperlakukan sebagai OLAP di *graph database*, karena itu mempengaruhi *graph global query* untuk model data yang tidak optimal[2].

OLAP pada database relasional, meskipun database relasional dirancang untuk OLTP, beberapa tahun ini banyak database relasional yang sudah memperoleh kemampuan OLAP. Hal ini dicontohkan melalui penambahan operator-operator khas OLAP seperti CUBE dan *roll-up* sebagai standar ANSI SQL. Operasi *multidimensional* OLAP pada database relasional biasanya melibatkan *roll-up*, *drill-down*, *slicing* and *dicing*, dan *pivoting*. Untuk alasan bisnis, ada kecenderungan yang jelas antara analisis *real time* atau analisis *near-real time*[2].

OLAP pada *graph database*, meskipun *graph* telah dipelajari secara substansial untuk jangka waktu yang lebih lama dibandingkan dengan database relasional telah ada. Saat ini *graph database* masih kurang mendukung untuk OLAP. Pada kenyataannya, sangat sering muncul kutipan-kutipan mengenai *graph database adoption*[2].

2.4 Graph OLAP Framework

2.4.1 Graph Model

Pada *Graph OLAP Framework* data dipresentasikan dengan koleksi dari *network snapshots* $G = \{G_1, G_2, \dots, G_n\}$, dimana setiap *snapshots* $G_i = (I_1, I_2, \dots, I_k; G_i)$. Pada $I_1, I_2, \dots, I_k$, k *informational attributes* menggambarkan *snapshots* secara keseluruhan dan $G_i = (V_i, E_i)$ adalah *graph*[3].

Dimensi dan ukuran adalah dua konsep yang merupakan fondasi dari OLAP[3]. Pertama, dimensi digunakan untuk membentuk

cuboid lattice dan mempartisi data ke sel lain. Dimensi berperan sebagai dasar dari *multidimensional* dan *multi-level analysis*. Kedua, ukuran dihitung untuk mengagregasi data yang dicakup untuk memberikan rangkuman dari data tersebut.

Pada gambar 2-3 terdapat dua dimensi, yaitu *venue* dan *tahun*. Peran dari dua dimensi ini adalah untuk mengorganisir *snapshot* menjadi sebuah grup yang memiliki perspektif berbeda[3].

2.4.2 Informational Dimensions

Pada pembahasan *graph model* sebelumnya sebuah set dari *informational attributes* $\{I_1, I_2, \dots, I_k\}$ merupakan *informational dimensions* dari *Graph OLAP*, atau dapat kita singkat sebagai Info-Dims[3].

2.4.3 I-Aggregated Graph

I-Aggregated Graph merupakan sebuah *attributed graph* yang dapat dihitung berdasarkan koleksi *network snapshots* $G' = \{G_1, G_2, \dots, G_n\}$ yang memiliki Info-Dims yang sama[3].

Graph pada gambar 2-3 menjelaskan frekuensi dari kolaborasi antar penulis pada sebuah konferensi pada suatu waktu. *Graph* tersebut merupakan salah satu contoh dari I-Aggregated Graph. Menurut chen [3] rangkuman dari operasi pada OLAP klasik yang diterapkan di *graph I-OLAP* adalah sebagai berikut:

- Roll-up: melapisi atau menindih beberapa *snapshots* untuk membentuk level yang lebih tinggi.
- Drill-down: mengembalikan ke level *snapshots* yang lebih rendah dari lapisan *aggregated graph* yang lebih tinggi.
- Slice/dice: memilih subset dari kualifikasi *snapshots* berdasarkan Info-Dims.

2.4.4 Localization

Pada I-OLAP *aggregated graph* M_l yang merangkum sekelompok *network snapshots* $G' = \{G_{i1}, G_{i2}, \dots, G_{iN}\}$, kita hanya perlu memeriksa relasi dari v pada $G_{i1}, G_{i2}, \dots, G_{iN}$ untuk mengkalkulasi *node* dari v di M_l dan kita hanya perlu untuk memeriksa lingkungan dari u, v pada $G_{i1}, G_{i2}, \dots, G_{iN}$ untuk menghitung *edge* dari (u, v) di M_l , maka penghitungan tersebut dapat dikatakan sebagai *localizable* [3].

2.5 Dataset

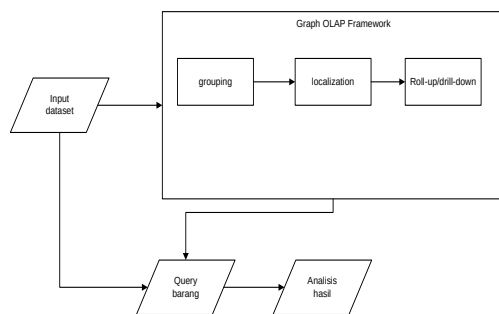
Dataset merupakan perwujudan data dalam memori yang memberikan model program relational yang konsisten tanpa memperhatikan asal dari sumber data. Digunakan Untuk mengatur sendiri query yang akan dijalankan dengan menggunakan DataAdapter dalam menggunakan parameter dalam pembuatan report [12].

DataSet merupakan pepresentasi data dalam memori yang memberikan model pemrograman relational yang konsiten, tanpa mmemperhatikan asal dari sumber data. Suatu data set mewakili serangkaian data secara lengkap termasuk table yang berhubungan, constain(aturan), dan relasi antar table [12].

3. Metode Penelitian

3.1 Deskripsi Sistem

Dalam perancangan sistem *graph database* dengan metode *graph OLAP framework* ini akan gambarkan gambaran umum dari pemodelan sistem yang dibangun dibawah ini.



Gambar 3.1 Gambaran Umum Sistem

Dalam tugas akhir ini diimplementasikan sistem *graph database* yang menggunakan metode *graph OLAP framework*. Di sistem ini akan diimplementasikan operasi-operasi OLAP pada database relasional seperti *roll-up* dan *drill-down*. Operasi *slice/dice* akan direpresentasikan di query pencarian barang. Pada *graph OLAP framework* terdapat sebuah fungsi yang bernama *localization*. *Localization* tersebut digunakan untuk menampilkan setiap frekuensi relasi antar node yang ada pada setiap *snapshots*. Sistem ini pun akan meranking tiap node dari tinggi ke rendah berdasarkan frekuensi kemunculan di dataset penjualan barang *outdoor* di tiap *snapshots*.

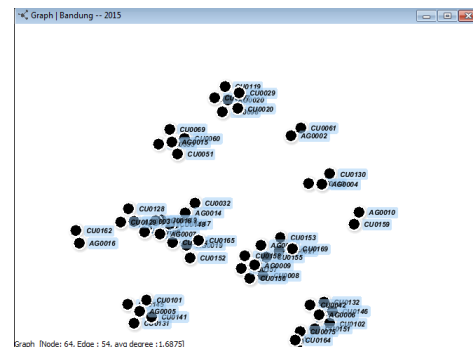
4. Pembahasan

4.1 Graph OLAP Framework

Seperti yang dijelaskan pada bab sebelumnya, *Graph OLAP framework* akan membentuk *snapshots* atau bisa disebut dengan sel dengan dimensi dan ukuran sebagai parameternya. *Snapshots* atau sel ini bertujuan untuk membentuk data yang *multidimensional* dan *multi-level analysis*. Pada gambar 3-1 terdapat empat fase pada *graph OLAP framework* yaitu *grouping*, *localization*, *roll-up/drill-down*, dan *output*. Fase-fase tersebut akan dijelaskan dibawah ini.

4.1.1 Grouping

Pada fase grouping ini data akan dibentuk menjadi sebuah *snapshots* dengan dimensi dan ukuran sebagai parameternya. Pengelompokan *snapshots* ini menggunakan dua dimensi yaitu kota dan tahun. Oleh karena itu, data yang direpresentasikan akan dikelompokkan berdasarkan kota dan tahun, contohnya [Jakarta,2013], [Jakarta,2014], [Bandung,2013], [Bandung,2014].



Gambar 4. 1 Snapshots [Bandung,2015]

4.1.2 Localization

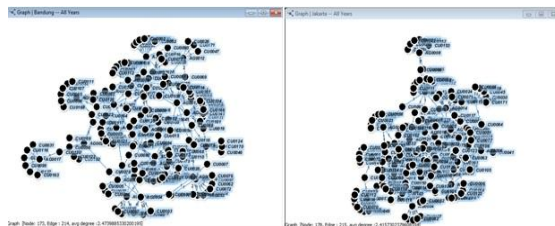
Pada fase *localization* sistem akan menampilkan frekuensi relasi antar node pada setiap *snapshots* yang telah dibentuk pada fase sebelumnya dan akan memberikan ranking terhadap node dari tinggi ke rendah berdasarkan frekuensi kemunculan di setiap *snapshots*. Namun, penulis membagi dua node yang diranking menjadi *node* agen dan *node* pelanggan dikarenakan jumlah frekuensi yang pasti akan sangat berbeda antara *node* agen dan *node* pelanggan.

[illegible]

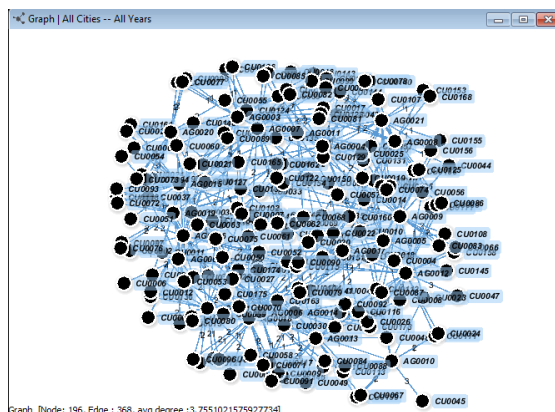
Gambar 4.2 gambar tabel localization

4.1.3 Roll-up/Drill -down

Pada fase *roll-up* dan *drill down* ini akan membentuk *multidimensional* dan *multi-level* analisis data dengan menerapkan *I-aggregated graph* operasi *roll-up* akan membentuk level *snapshots* data yang lebih tinggi. Contohnya, pada fase *grouping* sebelumnya terdapat empat *snapshots* yaitu [Jakarta,2013], [Jakarta,2014], [Bandung,2013], [Bandung,2014]. Setelah melakukan *roll-up*, *snapshots* yang dihasilkan adalah [Jakarta,all-years] dan [Bandung,all-years]. Proses *roll-up* itu sendiri dilakukan dengan menindih antara beberapa *snapshots*. Apabila terdapat node yang memiliki Info-Dims yang sama jumlah frekuensi antar node tersebut akan bertambah. Sebaliknya dengan *roll-up*, *drill-down* merupakan operasi yang mengembalikan *snapshots* dari level yang lebih tinggi ke level yang lebih rendah. Terdapat dua level pembentukan *snapshots* pada sistem ini seperti yang dijelaskan sebelumnya pada level 1 pembentukan *snapshots* menjadi [Bandung,all-years] dan [Jakarta,all-years]. Level selanjutnya *snapshots* akan menjadi [all-cities,all-years]. Berikut gambaran dari *snapshots* yang telah di *roll-up*.



Gambar 4.3 Snapshots roll-up level 1



Gambar 4.4 snapshots roll-up level 2

4.2 Query Barang

Pada proses *query* barang. Pencarian barang dilakukan dengan memasukkan kode barang dan waktu pembelian. Untuk parameter waktu pencarian barang dapat dibagi berdasarkan tanggal, bulan, dan tahun.

Dengan perangkat lunak ini kita akan mendapatkan *Graph Database Management System* yang sudah menerapkan beberapa fungsi OLAP klasik. Adapun tujuan dilakukannya pengujian pada bab ini adalah:

1. Mendapatkan data yang bersifat *multidimensional* dan *multilevel view*.
2. Mendapatkan frekuensi *edge* antar *node* dan peringkat frekuensi kemunculan *node* dari tinggi ke rendah.
3. Mengetahui apakah operasi *roll-up*, *drill-down*, dan *query* pencarian barang berhasil diterapkan.

Seperti yang telah dijabarkan pada bagian pengujian khususnya pada bagian *grouping* dan *roll-up/drill-down*. Sistem perangkat lunak ini telah dapat membentuk data yang bersifat *multidimensional* dan *multi-level view* sehingga poin pertama dari tujuan pengujian telah tercapai.

Pada bagian pengujian *localization*, sistem telah dapat menampilkan peringkat kemunculan *node* dan juga frekuensi *edge* antar *node* pada setiap *snapshotsnya*. Sehingga poin kedua dari tujuan pengujian ini telah tercapai.

Point terakhir dari tujuan pengujian ini adalah mengetahui apakah operasi khas OLAP klasik dan *query* pencarian barang telah berhasil diterapkan. Berdasarkan pengujian sebelumnya operasi-operasi khas OLAP tersebut dan juga pengujian *query* pencarian barang telah berhasil di terapkan pada sistem ini.

5. Kesimpulan dan Saran

5.1 Kesimpulan

Setelah penelitian dan analisis dilakukan, dapat ditarik beberapa poin kesimpulan antara lain :

1. Sistem dapat menerapkan OLAP pada *graph database* sesuai dengan *graph OLAP framework*, yaitu berhasil merepresentasikan data secara *multidimensional* dan *multi-level view*.
2. Operasi khas-khas OLAP pada database relasional berhasil diterapkan seperti *roll-up* dan *drill-down*.
3. Operasi *localization* dimana menampilkan frekuensi antar node dan pemberian peringkat dari tinggi ke rendah berdasarkan frekuensi kemunculan pada *snapshots* juga berhasil diterapkan.
4. *Query* pencarian barang berhasil diimplementasikan.

5.2 Saran

Berikut ini adalah beberapa saran yang dapat dijadikan pedoman untuk melakukan pengembangan tugas akhir ini, antara lain :

1. Semoga kedepannya penelitian OLAP pada *graph database* akan semakin berkembang dan dapat menerapkan sepenuhnya OLAP pada *graph database*.
2. Mencoba menerapkan beberapa framework OLAP lain

Daftar Pustaka

- [1] A. Soshani, "OLAP and Statistical Databases: Similarities and Differences" Berkeley: Lawrence Berkeley National Laboratory.
- [2] Michal Bachman, *GraphAware: Towards Online Analytical Processing in Graph Databases*, London: Imperial College London, 2013.
- [3] C. Chen, X. Yan, F. Zhu, J. Han, and P. S. Yu, "Graph OLAP: Towards Online Analytical Processing on Graphs", ICDM'08, 103-112, Pisa, Italy, Dec. 2008.
- [4] C. Chen, X. Yan, F. Zhu, J. Han, and P. S. Yu, "Graph OLAP: A Multi-Dimensional Framework for Graph Data Analysis", KAIS, Volume 21, Issue 1, 41-63, 2009.
- [5] C. Li, P. S. Yu, L. Zhao, Y. Xie, W. Lin, "InfoNetOLAPer: Integrating InfoNetWarehouse and InfoNetCube with InfoNetOLAP".
- [6] D. Suryadi, N. Priatna, "PAMA4208/MODUL1".
- [7] Harris Febryantony, *Implementasi Prinsip MDL untuk Kompresi Graph Database Menggunakan Algoritma Greedy*, 2015.
- [8] I. Robinson, J. Webber, E. Eilfrem. 2013. *Graph Database*. United States of America: O'Reilly Media.
- [9] K. U. Khan, K. Najeebullah, W. Nawaz, Y. Lee, "OLAP on Structurally Significant Data in Graphs" Republic of Korea: Kyung Hee University.
- [10] P. Zhao, X. Li, D. Xin, and J. Han. Graph Cube: On Warehousing and OLAP Multidimensional Networks. In Proc. of ACM SIGMOD, 2011, pp 853-864.
- [11] M. Berlingerio, M. Coscia, F. Giannotti, A. Monreale, D. Pedreschi "Multidimensional networks: foundations of structural analysis" 2012.
- [12] <http://ajigatotsantosa.blogspot.com/2011/02/pengertian-objek-objek-pada-visual.html> diakses pada 20 oktober 2014