Optimasi Datawarehouse Menggunakan MaterializedView dan Framework OVSP (Studi Kasus : PT. Genesis Indojaya Land)

M Baitur Ridwan[1], Ade Saputra[2], Adhie Sastro Sadewo[3], Probo Kusumo[4]
[1,2,3] Universitas Budi Luhur, Jl. Ciledug Raya Petukangan Utara – Jakarta Selatan
[4]Universitas Indraprasta PGRI, Jl. Nangka Raya No.58 C, Tanjung Barat, Jakarta Selatan
E-mail: [1]baitur.ridwan27@gmail.com, [2]234tkj@gmail.com, [3]adhie.sastro@gmail.com, [4]bow.itsm@gmail.com

Abstrak

Penggunaan Data Warehouse sebagai salah satu instrumen penting dalam pengambilan keputusan semakin dirasakan manfaatnya, sehingga banyak digunakan disemua perusahaan baik skala menengah maupun perusahaan besar. Jumlah data yang terus meningkat dari hari ke hari menjadikan tantangan tersendiri untuk dapat menyajikan laporan yang cepat dan akurat. MaterializedView Selection merupakan salah satu teknik yang dapat digunakan untuk mengurangi response time yang lambat ketika terjadi akses pada laporan dengan data yang besar maupun query yang kompleks. Penelitian ini bertujuan untuk meningkatkan waktu respon menjadi lebih baik dari kondisi sebelumnya dengan dengan mengggunakan framework OVSP yang di kembangkan. Pembentukan MaterializedView pada semua query yang ada adalah tidak mungkin dikarnakan beban maintenance dan penyimpanan yang menjadi semakin bertambah. Penelitian ini melakukan strategi seleksi pada MaterializedView dalam sebuah Data Warehouse dengan memperhatikan komposisi aspek maintenance, frekuensi query, waktu respon query dan penyimpanan. Hasil pengujian menggunakan framework OVSP atau OptimizeView Selection Problem yang terlah di kembangkan dalam penelitian ini terbukti menghasilkan komposisi MaterializedView secara otomatis dan terjadi peningkatan waktu respon yang baik rata-rata sebesar 65% pada hasil pengujian yang dilakukan.

***Kata kunci** : Datawarehouse; Materialized view; Query*

Abstract

The use of Data Warehouse as one of the important instruments in making decisions that are accepted Benefits, so that it is used by many companies both medium and large companies. The amount of data that continues to increase from day to day makes it a special challenge to be able to make reports that are fast and accurate. RealizedView Selection is one of the techniques that can be used to reduce response times that are increasingly complicated to occur in reports with large data or complex queries. This study aims to improve response times better than before using the developed OVSP framework. The establishment of a manifestation of the appearance of all existing requests is not possible due to increased maintenance and storage costs. This study selects strategies for displaying data in the Data Warehouse by considering the composition of maintenance, frequency queries, time response queries, and storage. The results of testing using the OVSP framework or Display Optimization Selection Problems that have been developed in this study are proven to produce automatic materialalized composition of views and an increase in response time is good by an average of 65% on the results of tests conducted.

***Keywords :** Datawarehouse; Materialized view; Query*

1. PENDAHULUAN

Pesatnya perkembangan teknologi informasi begitu sangat cepat searah dengan tantangan kebutuhan bisnis pada perusahaan yang semakin kompleks. Berkembangnya bisnis perusahaan menjadikan data transaksi operasional menjadi semakin besar jumlahnya. *Data Warehouse* adalah kumpulan dari berbagai sumber data yang jumlahnya dapat mencapai lebih dari puluhan juta record dan bersifat *time-consuming* pada pemrosesan *query* didalamnya[1].

Transaksi pembelian properti baik dalam bentuk uang muka yang dicicil, pembayaran akad kredit maupun transaksi biaya-biaya operasional lainnya pada perusahaan properti adalah merupakan hal yang mendasar yang menjadi proyeksi utama maupun tolak ukur bagi terciptanya kondisi neraca keuangan yang sehat bagi perusahaan property pada umumnya. Jumlah *customer*, transaksi properti yang banyak dari hari ke hari menjadikan jumlah record terus meningkat pada *Data Warehouse* di PT.Genesis Indojaya Land. Sedangkan kebutuhan untuk akses report maupun query dalam catatan waktu yang masih bisa diterima secara umum dari *Data Warehouse*, sangatlah penting yang salah satunya berfungsi sebagai alat pengambilan keputusan.

Data Warehouse merupakan kumpulan jumlah data yang sangat besar dan berasal dari berbagai macam sumber data untuk kemudian di integrasikan menjadi satu kesatuan dalam sebuah *framework*[2]. *Data Warehouse* juga sering di definisikan sebagai sebuah *repository* atau tempat penyimpanan yang berisi informasi yang terintegrasi dari berbagai sumber yang dapat dianalisa[3]. Pada umumnya *Data Warehouse* bersifat *non-volatile, time-variant, integrated* dan *subject-oriented* yang mana dapat membantu dalam proses pengambilan keputusan oleh kalangan tertentu seperti misalnya manajemen pada sebuah perusahaan.

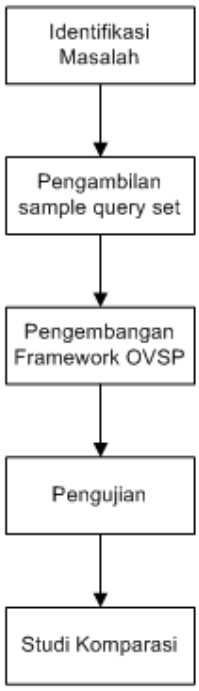
Materialized View (MV) merupakan sebuah *View* yang dimaterialisasikan artinya secara fisik MV itu ada dalam sebuah database layaknya obyek tabel dengan cara menyimpan hasil relasi kedalam *database*. Secara definisi MV adalah struktur fisik yang melakukan perhitungan awal untuk menghasilkan data sehingga akan meningkatkan waktu akses data [4]. Oleh karena itu akan ada konsenkuensi bahwa akses database akan menjadi lebih cepat ketika mengakses MV dibandingkan dengan cara akses melalui *view*. MV menghilangkan proses untuk melakukan komputasi ulang terhadap sebuah query seperti yang terjadi pada*View* [5].

Algoritma atau *framework* OVSP dimana secara garis besar algortima ini melakukan perhitungan seleksi *query* analitik yang ada untuk di materilaisasikan berdasarkan bobot masing-masing yang terdapat pada *query* set dan storage space [4], kemudian dilakukan juga perhitungan *query access cost* dan *maintenance cost*, selanjutnya total cost dari tiap *View* dihitung dan *View* dengan cost yang perhitungannya sesuai dengan *maintenance* dan *space constrain* dipilih untuk proses materialisasi [6].

Algoritma atau *framework* OVSP (*Optimized View Selection Problem*) diperkenalkan oleh [4] yang setelah sebelumnya diawali dengan *framework* CEMS [7] dan juga OCEMS [8] dimana secara garis besar algortima ini melakukan perhitungan seleksi *query* analitik yang ada untuk di materilaisasikan berdasarkan bobot masing-masing yang terdapat pada *query set* dan *storage space*, kemudian dilakukan juga perhitungan *query access cost* dan *maintenance cost*, selanjutnya total *cost* dari tiap *View* dihitung dan *View* dengan *cost* yang perhitungannya sesuai dengan *maintenance* dan *space constrain* dipilih untuk proses materialisasi [4].

2. METODE PENELITIAN

Langkah-langkah atau desain penelitian akan direncanakan dalam rangka melakukan penelitian sesuai dengan kerangka konsep atau pola pikir pemecahan masalah. Gambar 1 berikut adalah desain penelitian :



Gambar 1. Desain Penelitian

1. **Identifikasi Masalah**
   Terdapat *response time* yang lambat saat user mengakses report maupun melakukan *query* analitik terkait data transaksi properti pada *Data Warehouse* perusahaan. Jumlah data yang kian hari terus bertambah dan kondisi perusahaan yang belum terbilang stabil, mendorong semua teknologi yang digunakan pada server harus lebih efektif dan efisien.
2. **Pengambilan sample berupa *query set***
   Sample data query set diambil dari data query analitik ke *Data Warehouse* yang dilakukan oleh semua user yang menggunakan.
3. **Pengembangan *Framework* OVSP**
   Pada tahapan ini akan di lakukan pembuatan pengembangan *framework* OVSP dengan menggunakan bahasa pemrograman Oracle PLSQL. Secara garis besar ada beberapa perhitungan yang digunakan seperti kalkulasi *query cost, maintenance cost* dan juga *storage cost.*
4. **Pengujian**
   Pada langkah ini akan dilakukan pengujian seperti *response time, maintenance cost storage cost*. Hasil akhir nya adalah berupa table yang berisi kombinasi dari ketiga perhitungan tersebut.
5. **Studi Komparasi**
   Pada bagian ini dilakukan proses komparasi antara performa sebelum dan sesudah implementasi seleksi *materialized view.*

Peneltian ini menggunakan beberapa metode untuk pengambilan data dari PT.Genesis Indojaya Land, antara lain :

1. Studi literatur melakukan studi literatur dari berbagai buku, arsip, dokumen ataupun artikel yang berkaitan dengan kebutuhan penelitian.
2. Metode observasi dilakukan dengan melihat, menggabungkan dan menganalisa proses yang terjadi pada PT.Genesis Indojaya Land dengan barbagai sumber data yang lain.

3. HASIL DAN PEMBAHASAN

3.1 Tahap Pengumpulan Data

Pengumpulan data dilakukan untuk menghasilkan informasi yang tepat dan menjawab permasalahan penelitian [9]. Metode pengumpulan data yang digunakan dalam penelitian ini adalah pengumpulan data primer. Data penelitian yang diperoleh secara langsung dari sumber aslinya yakni data yang digunakan adalah data *query set* (*query history*) atau *query* analitik pada PT.Genesis Indojaya Land untuk mengakses ke *Data Warehouse*. Database yang digunakan adalah oracle dengan mengakses table histori v$sql seperti pada gambar 2 berikut ini:

ELAPSED_TIME	ROWS_PROCESSED	DISK_READS	SQL_FULLTEXT
4150065	26	29981	WITH OBICOMMON0 AS (select DENSE_RAN
4140240	20	29981	WITH OBICOMMON0 AS (select DENSE_RAN
4173092	21	29981	WITH OBICOMMON0 AS (select DENSE_RAN
1315657	20347	0	WITH SAWITH0 AS (select sum(T46195.'
34515024	13	1	WITH OBICOMMON0 AS (select DENSE_RAN
1837759	2	0	WITH OBICOMMON0 AS (select T46530.CA
1882362	2	0	WITH OBICOMMON0 AS (select T46530.CA
125107	2	0	WITH OBICOMMON0 AS (select T46530.CA
1933429	52	0	WITH OBICOMMON0 AS (select T46530.CA
527633	1	0	SELECT SUM(TOTAL) FROM FACT_LBK

Gambar 2. Query Set

Pengambilan sample dilakukan pada 10 query set dengan berbagai kondisi jumlah data. Setelah itu akan dilakukan pengujian sebelum dan sesudah dilakukan optimasi.

3.2 Analisis Sistem Yang Berjalan

Analisis sistem yang berjalan adalah melakukan pengujian atau perhitungan untuk mengukur seberapa besar response time beserta cost matrik yang diperlukan pada sample data yang akan digunakan. Pada bagian ini akan dilakukan pengujian dan perhitungan *cost* pada 10 *query set* atau query analitik yang ada pada PT.Genesis Indojaya Land untuk melihat kondisi sistem yang berjalan. Selain itu akan dilihat juga seberapa lama respon yang di perlukan pada setiap *query* yang kemudian akan dilakukan komparasi dengan *framework* yang akan dikembangkan.

Pada bagian ini *maintenance cost* dan *storage cost* akan dihitung dengan asusmsi angka 0 atau 1 hal ini disebabkan oleh tidak adanya beban maintenance serta *storage cost* apabila tidak menggunakan *materialized view*, akan tetapi *response time* yang lambat akan menjadikan total *cost* menjadi membesar dikarnakan harus melakukan *full table scan* pada query analitik yang ada.

Tabel 1. Hasil eksekusi menggunakan *framework* OVSP

Query Name	Response Time	Frequency	Number of Base Table	is Materialized
Q1	55,278,627	9	3	No
Q2	924,075	8	3	No
Q3	43,624	2	2	No
Q4	7,302	4	1	No
Q5	59,724,006	2	1	No
Q6	78,854,526	1	2	No
Q7	72,868,863	6	3	No

Query Name	Response Time	Frequency	Number of Base Table	is Materialized
Q8	65,066,002	6	2	No
Q9	59,191,593	8	2	No
Q10	65,630,000	5	2	No

Dari tabel 1 diatas terlihat bahwa respon time tercepat diantaranya terdapat pada Q2, Q3 dan Q4 yang memiliki base table yang cukup variatif. Pada bagian kolom is materialized tidak ada yang termaterialisasi hal ini dikarnakan sistem berjalan masih belum menggunakan *framework* OVSP.

3.3 Pengujian Framework OVSP

Pada penelitian ini akan dilakukan pengujian optimasi *Data Warehouse* dengan *framework* OVSP dengan kondisi infrastruktur dan kompleksitas sistem yang berbeda dari penelitian sebelumnya. Penelitian ini akan melakukan pengujian dengan menggunakan 10 sample query set dalam *Data Warehouse* pada PT.Genesis Indojaya Land yang terdiri dari delapan *table* dimesin dan 2 buah table fakta.

Sample data yang digunakan adalah sama dengan pengujian yang dilakukan sebelumnya yaitu pada sistem yang berjalan atau sebelum implementasi *framework* OVSP agar dapat di ukur dan dibandingakan. Jumlah data pada tabel fakta terdiri dari empat puluh juta record lebih, sehingga dengan demikian maka harapannya pada penelitian ini dapat di uji seberapa baik *framework* OVSP bekerja pada kondisi data yang cukup banyak.

Selanjutnya dari informasi cost matrix pada tiap query set diatas akan dilakukan perhitungan menggunakan rumus (1) teori *framework* OVSP seperti di bawah ini :

$$Total\ Cost = QP_{cost} + VM_{cost} \quad \text{..... (1)}$$

Query processing cost pada rumus (1) di dapatkan dari frequensi dikalikan dengan *query access* atau *query response time* pada tiap subquery yang ada, dimana *Freq* adalah frequensi *query* dan *Ca(V)* adalah merupakan *response time* pada sebuah *view*. Seperti persamaan (2) berikut :

$$QP_{cost} = \frac{1}{\sum_{i=1}^{N} Freq}\ x\ Ca(V) \quad \text{..... (2)}$$

Total cost di hasilkan dari penjumlahan antara *query processing cost* dengan View *maintenance cost. Query processing cost* didapat dari rumus perhitungan *query processing* sebelumnya sedangkan *View maintenance cost* merupakan beban maintenance yang dibutuhkan yang di dapat dari total update frekuensi pada table fisik seperti table dimensi, semakin sering perubahan yang terjadi pada sebuah tabel dimensi maka akan semakin besar pula beban maintenance yang dibutuhkan.

Setelah itu dilakukan perhitungan terhadap total cost pada semua query set yang ada. Dengan menggunakan rumus Total *cost* yaitu total *query processing cost* di tambah dengan *View maintenance cost.*

Tabel 2. Hasil perhitungan dengan OVSP

Query	Response Time	Freq	is Materialized	Storage Cost	Maintenance Cost	QPCost	TCost
Q1	7,113,748	9	Yes	10	2	0.111	2.111
Q2	302,441	8	No	10	2	1.125	3.125

Query	Response Time	Freq	is Materialized	Storage Cost	Maintenance Cost	QPCost	TCost
Q3	43,219	2	No	10	2	4.5	6.5
Q4	5,063	4	No	10	2	1.5	3.5
Q5	12,281,073	2	Yes	10	2	1	6
Q6	52,271,738	1	Yes	10	2	4	6
Q7	16,114,700	6	Yes	10	4	0.5	4.5
Q8	56,957,553	6	Yes	10	4	0.75	4.75
Q9	18,588,500	8	Yes	3	4	1	5
Q10	21,189,641	5	Yes	3	2	1	5

Dari tabel 2 di atas dapat dilihat bahwa tidak semua *query set* dibentuk menjadi *materialized view*, karna hal ini bergantung pada hasil perhitungan dan nilai *threshold* atau nilai ambang batas yang termasuk kedalam *query set* yang layak untuk dikonversi menjadi *materialized view*. Berbeda pada perhitungan *cost* dari sistem yang berjalan, salah satunya *maintenance* dan *storage cost* yang cukup variatif. Semakin detail data yang disimpan menjadi *Materialized View* maka akan semakin besar kapasitas penyimpanan atau storage yang dibutuhkan [10]. Sedangkan semakin banyak table yang digunakan dalam suatu *query set* maka akan semakin besar pula beban *maintenance* yang diperlukan. Pengujian dari sisi query frequency akan di tentukan secara random, hal ini untuk melihat seberapa baik *framework MaterializedView* bekerja diberbagai frekuensi query.

3.4 Hasil Komparasi

Berikut adalah hasil komparasi atau perbandingan dari sisi waktu respon pada saat melakukan query analitik. Perbandingan dilakukan pada kondisi sistem yang berjalan dimana belum menggunakan materialisasi dan kemudian dibandingkan dengan setelah menggunnakan *framework* OVSP (*Optimized View Selection Problem*).

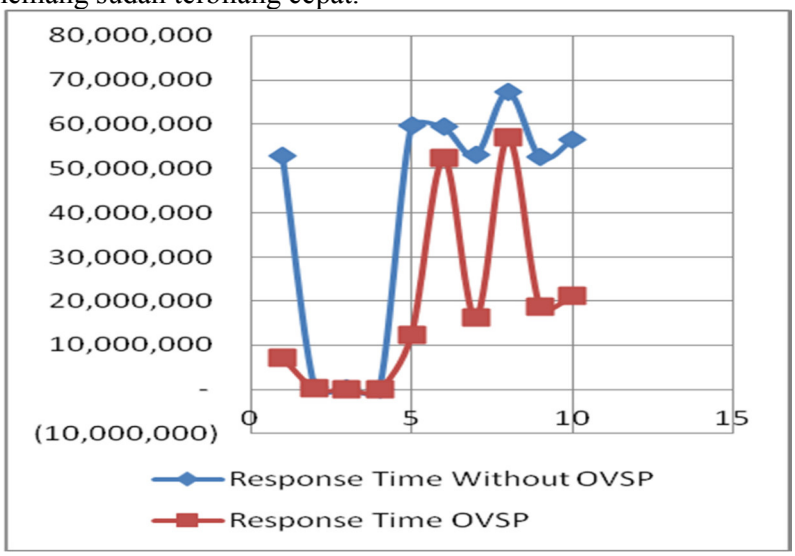
Tabel 3 Hasil Komparasi *Framework* OVSP Dengan Sistem Berjalan

Query Name	Freq	is Materialized	QPCost	TCost	Response Time Without OVSP	OVSP Response Time	% Increase
Q1	9	Yes	0.111	2.111	52935184	7,113,748	644.13 %
Q2	8	No	1.125	3.125	435021	302,441	43.84 %
Q3	2	No	4.5	6.5	211569	43,219	389.53 %
Q4	4	No	1.5	3.5	3758	5,063	25.78%
Q5	2	Yes	1	6	59649312	12,281,073	385.70 %
Q6	1	Yes	4	6	59409137	52,271,738	13.65 %
Q7	6	Yes	0.5	4.5	53147667	16,114,700	229.81 %
Q8	6	Yes	0.75	4.75	67324130	56,957,553	18.20 %
Q9	8	Yes	1	5	52684746	18,588,500	183.43 %
Q10	5	Yes	1	5	56428902	21,189,641	166.30 %

Rata-rata Peningkatan =
(644.13 + 43.84 + 389.53 + (25.78) + 385.70 + 13.65 + 229.81 + 18.20 + 183.43 + 166.30) / 10 = 65 %

Berdasarkan tabel 3 diatas, hasil komparasi *framework* OVSP dengan sistem berjalan, terjadi peningkatan waktu respon yang baik rata-rata sebesar 65% yang disertai dengan

pembentukan 7 *Materialized View* dan 3 query tidak di materialisasikan dikarnakan waktu respon yang memang sudah terbilang cepat.



Gambar 3. Perbandingan Waktu Respon

Pada gambar 3 di atas dapat diterlihat bahwa waktu eksekusi *query* akan menjadi sedikit selisih waktunya dikarnakan pada ada beberapa query dengan waktu respon yang asalnya rendah serta total costnya lebih rendah dibanding yang lain, seperti yang telah disinggung sebelumnya hal ini menjadikan materialisasi tidak akan terbentuk dari *query* dengan waktu respon cepat. Berdasarkan hasil grafik komparasi antara kondisi sistem berjalan dengan setelah penggunaan *framework* ovsp pada datawarehouse PT.Genesis Indojaya Land, terjadi peningkatan waktu respon yang baik rata-rata sebesar 65%.

4. KESIMPULAN

Strategi dan komposisi pembentukan *Materialized View* pada *Data Warehouse* yang tepat, dapat secara efektif mempercepat proses akses *query* dengan beban pemeliharaan dan konsumsi storage yang rendah, hal ini dapat dilihat ketika *query set* yang memiliki cost yang lebih rendah atau waktu respon lebih cepat tidak termaterialisasi oleh *framework* ini sehingg meminimalisir pembentukan *Materialized View* yang tidak perlu. Penerapan algoritma atau *Framework* OVSP (*Optimized View Selection Problem*) dengan database oracle pada PT.Genesis Indojaya Land mampu meningkatkan kecepatan 65% akses query analitik ke *Data Warehouse*.

DAFTAR PUSTAKA

[1] M. K. Sohrabi and H. Azgomi, "TSGV: A Table-Like Structure-Based Greedy Method For *MaterializedView* Selection In *Data Warehouse*s," *Turkish J. Electr. Eng. Comput. Sci.*, vol. 25, no. 4, pp. 3175–3187, 2017.

[2] M. K. Sohrabi and V. Ghods, "*MaterializedView* Selection for a *Data Warehouse* Using Frequent Itemset Mining," *J. Comput.*, vol. 11, no. 2, pp. 140–148, 2015.

[3] C. A. Dhote and M. S. Ali, "*MaterializedView* Selection in Data Warehousing," 2007.

[4] B. Ashadevi and D. A. Affairs, "A *Framework* for the*View* Selection Problem in Data Warehousing Environment," vol. 02, no. 09, pp. 2820–2826, 2010.

[5] Gupta, "Selection of views to materialize in a *Data Warehouse*," 2005.

[6] J. Kavita and S. Darudito, "Model Seleksi *MaterializedView* Untuk Meningkatkan Performansi Query Pada *Data Warehouse*.pdf." Binus University Graduate Program, Jakarta Barat, 2012.

[7] R. Balasubramanian and B. Ashadevi, "Cost Effective Approach for *MaterializedView* Selection in Data Warehousing Environment," *Int. J. Comput. Sci. Netw. Secur.*, vol. 8, no. 10, pp. 236–242, 2008.

[8] B. Ashadevi and R. Balasubramanian, "Optimized Cost Effective Approach for Selection of Materialized Views in Data Warehousing," *J. Comput. Sci. {\&} Technol.*, vol. 9, no. 1, pp. 21–26, 2009.

[9] M. S. Prof. Dr. H. Mudjia Rahardjo, "Metode Pengumpulan Data Penelitian Kualitatif," 2011. .

[10] T. Connoly and Carolyn Berg, *Pearson.Database.Systems.A.Practical.Approach.to.Design.Implementation.and.Management.6th.Global.Edition.1292061189.pdf*. 2009.