Implementasi ETL (*Extract, Transform, Load*) Pangkalan Data Perguruan Tinggi dengan Menggunakan *State-Space Problem*

Ramanti Dharayani , Kusuma Ayu Laksitowening, ST, MT [2], Amarilis Putri Yanuarfiani, ST,MT [3]
[1,2,3] Program Studi Sarjana Teknik Informatika, Fakultas Teknik Informatika, Universitas Telkom, Bandung
[1] dharayanii@gmail.com, [2] kal@ittelkom.ac.id, [3] amarilis.fiani@gmail.com

***Extraction, Transformation, dan Load* (ETL) adalah salah satu proses pada *datawarehouse*. Proses dari ETL adalah mengumpulkan data dari berbagai macam sumber. ETL adalah proses untuk mengolah data menjadi data yang bersih sesuai dengan ketentuan *datawarehouse*. Proses ETL pada umumnya terdiri dari berbagai macam aktivitas dan membutuhkan waktu serta memori yang cukup besar. Pada tugas akhir ini akan dilakukan implementasi ETL dengan menggunakan alur kerja *state space problem* pada kasus Pangkalan data perguruan tinggi. State space problem digunakan untuk menggambarkan alur proses ETL dan mencari urutan aktivitas pada proses ETL. Dari hasil pengujian ETL dilakukan perubahan urutan aktivitas dengan menggunakan transisi graf dan didapatkan hasil yang lebih optimal.**

Keywords*— *Extract, Transform, Load, ETL, state space problem, data warehouse, oracle warehouse builder

I. PENDAHULUAN

Pangkalan Data Perguruan Tinggi (PDPT) merupakan sumber informasi untuk sistem penjaminan mutu perguruan tinggi di Indonesia. PDPT memiliki kegiatan pengumpulan, pengolahan data dan informasi mengenai penyelenggaraan pendidikan tinggi pada seluruh perguruan tunggi oleh Ditjen Dikti. Kegiatan tersebut digunakan untuk mengawasi penyelenggaraan pendidikan tinggi oleh pemerintah. Untuk memudahkan kegiatan pengumpulan, pengolahan data dan informasi mengenai penyelenggaraan pendidikan tinggi oleh Dikti, diperlukan adanya perancangan informasi strategis seperti CIF (*Corporate Information Factory*) untuk menunjang kegiatan evaluasi dan perencanaan oleh kepala institusi untuk meningkatkan mutu institusinya.

CIF merupakan struktur ekosistem informasi yang dirumuskan oleh W.H Inmon. Menurut Inmon CIF adalah arsitektur logis yang bertujuan untuk menghasilkan kemampuan intelegensia bisnis dan manajemen bisnis yang berasal dari data yang di hasilkan operasional bisnis perusahaan. Diantara komponen arsitektur CIF, *datawarehouse* merupakan titik pusat integrasi data-tahap awal pengelolaan data menjadi informasi[1]. *Datawarehouse* merupakan suatu proses yang melibatkan tiga proses besar yaitu *basic* bisnis proses, *ETL(Extract, transform, Load)* dan dimensi bisnis.

Pada proses *datawarehouse* pangkalan data perguruan tinggi(PDPT), dibutuhkan adalah data historis dan bukan data terbaru. Data historis dan data terbaru bisa terletak di berbagai *data source* ataupun *database* yang berbeda. *Data source* yang terdiri dari berbagai macam dapat menimbulkan data yang terduplikasi maupun memiliki format data yang berbeda. Untuk menangani perbedaan format ataupun data yang terduplikasi dibutuhkan proses *ETL(Extract, Transform, Load)* agar data akhir berkualitas dan pada saat data masuk ke dalam *datawarehouse*, data dalam format yang sama. Proses ETL bertujuan untuk mentransformasikan data sesuai dengan ketentuan pada kebutuhan *datawarehouse*. Proses ETL melibatkan *data source* yang bisa lebih dari satu. Pada saat menjalani proses ETL transformasi yang dibutuhkan bisa satu kali aktivitas transformasi atau lebih untuk mendapatkan hasil data akhir yang berkualitas. Pada proses ETL pada umumnya memiliki beberapa permasalahan diantaranya proses ETL membutuhkan waktu eksekusi yang cukup lama, proses ETL juga membutuhkan memori yang besar dan terjadinya perubahan struktur data pada ETL menyebabkan harus mengulangi proses ETL. Adanya proses ETL yang membutuhkan waktu yang cukup lama dan memori yang besar terdapat beberapa pilihan urutan aktivitas yang akan mengurangi waktu, cost dan memori pada proses ETL. Untuk menggambarkan aktivitas proses ETL digunakan state space problem untuk mengubah urutan aktivitas dilakukan transisi graf pada desain state space problem yang telah dirancang sebelumnya.

Pada proses ETL karena bisa terdapat satu atau lebih aktivitas, sebaiknya direpresentasikan dengan *state space problem*. State space problem merupakan salah satu formula umum dari aksi intelegensi ruang permasalahan. State berisikan informasi dari proses yang menimbulkan effek untuk sebuah aksi menentukan next state dan final state[6]. State space problem merupakan kumpulan dari set yang memiliki konfigurasi dan memiliki solusi permasalahan yang dicapai. Proses pada .ETL pada pangkalan data perguruan tinggi bisa menyangkut banyak *input* data yang di proses dan merupakan proses yang cukup kompleks untuk itu sebaiknya menggunakan metode analisis *state-space problem*. *State space problem* merupakan metode analisis yang digunakan untuk menggambarkan sebuah proses secara detail dengan tujuan mendokumentasikan setiap proses ETL. suatu proses ETL yang cukup kompleks dapat digambarkan dengan *state space problem* dengan menggambarkan aktivitas sebagai graf agar proses ETL lebih terstruktur. Dengan adanya proses ETL yang terstruktur dapat mempermudah mendeteksi suatu kesalahan pada proses tersebut sehingga segala perbaikan dapat segera ditangani. Pada *state space problem* segala aktivitas digambarkan dengan bentuk graf sebagai *node* dan arah proses sebagai *edge*. Agar mendapat hasil yang optimal *node* dapat dilakukan transisi yang disebut dengan transisi graf. *Input* atau masukan dari *data source* yang digunakan untuk memenuhi kebutuhan *datawarehouse* PDPT merupakan data yang cukup kompleks karena bisa terdiri dari berbagai macam sumber data. Kompleksnya data tersebut bisa dirancang dengan

menggunakan metode analisis *state-space problem* agar segala aktivitas dapat dilihat secara detail proses-proses apa saja yang dilakukan untuk melakukan ETL pada *datawarehouse* PDPT.

II. LANDASAN TEORI

Berikut merupakan teori yang digunakan dalam implementasi ETL.

A. Data Warehouse

Data Warehouse secara istilah adalah sebuah sistem yang dapat mengambil dan mengkonsolidasikan data secara berkala dari berbagai sumber ke dalam sebuah dimensi *data store* yang telah di normalisasikan. *Data warehouse* bisa menyimpan *history* dari sebuah data dengan *query* yang akan digunakan ke dalam *business Intelegence* atau aktivitas analisis lain. [7]

Data warehouse adalah sebuah *relational database* yang di rancang untuk implementasi dari query dan analisa proses transaksi. I*data warehouse* berisikan data histori yang berasal dari data transaksi. Data yang terdapat pada *data warehouse* bisa juga berasal dari sumber lain yang berbeda misal dari file .sql, .xls, .txt dll. Data yang berasal dari berbagai macam sumber tersebut dapat diintegrasikan dan bisa memiliki hubungan satu dengan yang lainnya.

Selain *relational database,* lingkungan dari data warehouse meliputi ekstraksi, transportasi, transformasi dan pemuatan (ETL) solusi, Online Analytical Process (OLAP), *client analytic tools,* dan aplikasi lain yang mengelola data dan memberikan pada pengguna bisnis. [3]

B. Dimensional Model

Dimensional model merupakan tabel yang berisikan struktur data dari target proses ETL. Dimensional model berisikan tabel dimensi dan tabel fakta. Tabel-tabel tersebut berada diantara bagian belakang dan bagian depan proses. Dimensional model merupakan pemetaan fisik dari langkah sebelum dari proses memindahkan tabel ke end-user environment dimensional model lebih dikenal dengan struktur data yang digunakan oleh end-user untuk query dan analysis. Dimensional model sangat stabil dalam adanya perubahan pada data dan mudah di mengerti oleh user, dimensional model berisikan struktur data dan relative cepat saat melakukan query untuk relational database secara umum. Dimensional model digunalan sebagai dasar untuk membangun cube pada OLAP. [3]

C. ETL

ETL (Extract, Transform, Load) adalah sistem dasar dari *data warehouse*. Rancangan ETL yang baik dari sistem ekstraksi sumber data, mengedepankan kualitas data dan standar yang konsisten, data dari sumber yang terpisah sesuai, sehingga dapat diintegrasikan sehingga memberikan format data untuk di representasikan. Sistem ETL merupakan aktivitas *backbone* yang tidak terlihat oleh pengguna akhir datawarehouse. ETL memenuhi 70 persen sumber daya yang dibutuhkan dalam implementasi dan pemeliharaan *data warehouse*.[3]

Extract, Transform dan Load juga merupakan kumpulan proses persiapan data dari OLTP (Online Transaction Process). ETL merupakan fase pemrosesan data dari sumber data masuk ke dalam *data warehouse*. Tujuan dari ETL adalah mengumpulkan, menyaring, mengolah dan menggabungkan data-data yang relevan dari berbagai sumber untuk disimpan ke *dalam data warehouse*.[2]

D. State Space Problem

State space akan dimodelkan dengan sebuah graf asiklik dengan dua buah elemen : node dan edge, seperti graf pada umumnya. Node pada graf akan mewakili semua aktivitas dan recordset. Recordset adalah segala jenisdata store yang dapat menyediakan flat record schema. Dalam penelitian ini jenis data store yang mungkin digunakan adalah tabel relasi dan record file. Sedangkan edge menunjukkan sebuah hubungan antara penyedia dan pemakai data, yang dapat menghubungkan antara recordset dan aktivitas ataupun aktivitas dengan aktivitas. Setiap node memiliki karakteristik berbeda bergantung pada schemata yang dimilikinya. Sebuah recordset pasti hanya akan memiliki satu buah schemata, yaitu input schemata atau output schemata, sedangkan aktivitas memiliki setidaknya dua schemata, minimal satu untuk setiap schemata. Sebuah input schemata bertanggung jawab untuk membawa record data masuk ke dalam aktivitas untuk pemrosesan, dan output schemata bertanggung jawab untuk membawa data menuju pemakai data berikutnya. Sebuah aktivitas yang hanya memiliki sebuah input schema disebut unary, sedangkan yang memiliki dua input schema disebut binary.

Asumsikan A adalah sekumpulan aktivitas, RS adalah sekumpulan recordset, dan Pr adalah sekumpulan provider relationship (hubungan penyedia data), sehingga dengan demikian graf alur kerja dapat dinotasikan G(V, E), dimana V = A ∪ RS dan E = Pr. Recordset dapat terbagi menjadi dua jenis : RS yang menjadi penyedia data (source), dinotasikan dengan RSs dan RS yang menjadi pemakai data (target), dinotasikan dengan RSt. Sebuah aktivitas A dinotasikan secara formal dengan A = (Id, I, O, S), dimana :

1) Id adalah sebuah identitas unik untuk aktivitas,
2) I adalah kumpulan input schemata,
3) O adalah kumpulan output schemata, dan
4) S adalah ekspresi dalam aljabar relasional atau fungsi yang menjelaskan tugas suatu aktivitas. Operator aljabar yang akan digunakan antara lain : selection (σ), projection (π), Cartesian product (×), join (⋈), aggregation (γ), ordering (Θ), union (∪), difference (–), dan function application (f).[8]

E. Konseptual Model

Tujuan pada bagian ini adalah untuk menyajikan model konseptual untuk aktivitas ETL. Tujuan nya adalah menggunakan entitas berorientasi objek untuk menangkap proses ETL. Langkah pertama adalah dengan mendefinisikan notasi grafis dan metamodel. Kemudian secara detail akan mendefinisikan semua entitas pada metamodel tersebut.yang terdiri dari satu *end node* dan satu *coordinator node*, cukup menggunakan transparent mode. Sedangkan untuk jaringan yang terdiri dari banyak *node,* penggunaan API mode adalah suatu keharusan. Berikut adalah penjelasan lebih lanjut mengenai *transparent mode* dan API mode.

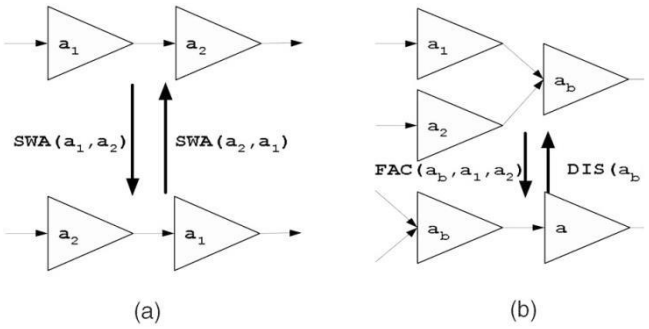
F. Logical Model

Logical model lebih difokuskan kepada aliran data dari data source menuju target atau data warehouse. Logical model lebih menjelaskan solusi teknis dari implementasi keseluruhan proses ETL. Sekenario keseluruhan ETL melibatkan aktivitas, sekumpulan record dan fungsi yang dapat digunakan bersamaan dengan grafik pada serial linier dan di eksekusi

secara berurutan yang disebut dengan architecture graph. Berikut merupakan notasi dari architecture graph.

G. Transisi graf

Transisi graf merupakan salah satu bagian yang penting dalam mendapatkan hasil yang optimal pada proses ETL. Transisi graf berfungsi untuk merubah susunan node pada graf sehingga dapat menghasilkan alur kerja yang lebih optimal. Transisi graf yang akan digunakan pada penelitian ini adalah sebagai berikut [2] :



Gambar 1 : Gambar transisis graf

1) *Swap* (Gambar 2-3 (a)).

Transisi ini dapat diterapkan pada dua buah aktivitas *unary* dengan merubah susunan kedua aktivitas tersebut. Susunan aktivitas A1.A2 akan berubah menjadi A2.A1 setelah di-*swap*, dan dinotasikan dengan SWA(A1, A2).

2) *Factorize* dan *distribute* (Gambar 2.3 (b)).

Factorize diterapkan dengan melibatkan sebuah aktivitas *binary* dan setidaknya dua aktivitas *unary* yang memiliki fungsionalitas sama namun pada alur yang berbeda, dan keduanya bertemu di aktivitas *binary*. Transisi *factorize* akan menggabungkan dua aktivitas *unary* tersbut yang kemudian akan diletakkan setelah aktivitas *binary*. Sedangkan *distribute* merupakan kebalikan dari *factorize*, dimana sebuah aktivitas *unary* akan dipecah menjadi dua atau lebih aktivitas dan diletakkan sebelum aktivitas *binary* pada alur yang berbeda. *Factorize* dan *distribute* ini sebenarnya merupakan transisi *swap* yang merubah aktivitas *unary* dan *binary*. Misalkan ada sebuah aktivitas *binary* Ab yang mendapatkan data dari A1 dan A2 dengan notasi (A1.Ab).(A2.Ab). Kemudian diterapkan *factorize* pada A1 dan A2 sehingga menjadi Ab.A(1+2), dimana A1 dan A2 digabungkan menjadi A(1+2) kemudian diletakkan setelah Ab. *Factorize* dinotasikan dengan FAC(Ab, A1, A2) dan *distribute* dinotasikan denagn DIS(Ab, A(1+2)).

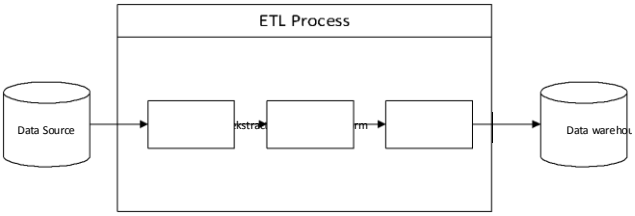
3) *Merge* dan *split* (Gambar 2.3 (c)).

Merge diterapkan dengan menyatukan dua buah aktivitas berbeda yang dapat disatukan berdasrkan *constraint* pada alur kerja ETL dan *split* diterapkan untuk memisahkan kembali aktivitas yang disatukan setelah aplikasi transisi selesai dijalankan. Misalkan aktivitas A1 dan A2 di-*merge*, maka akan menjadi A(1+2), dengan notasi MER(A(1+2), A1, A2). Begitu juga sebaliknya dengan *split*, notasinya menjadi SPL(A(1+2), A1, A2).

III. PERANCANGAN DAN IMPLEMENTASI

A. Gambaran Umum Sistem

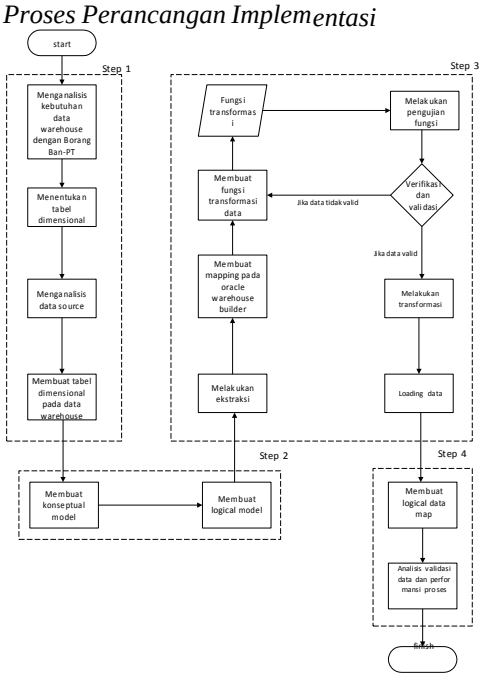
Sistem yang akan dibangun pada penelitian ini adalah proses ekstrasi, transformasi, dan loading data dari data source ke dalam data warehouse.



Gambar 2 : Diagram Blok Sistem

Gambar 2 merupakan spesifikasi dari sistem ETL proses yang dilakukan pada penelitian ini. Proses tersebut adalah mengambil data dari data source untuk di ekstrak ke dalam proses ETL, setelah data di ekstrak data di transformasikan sesuai dengan requirement yang di butuhkan. Setelah data selesai di transformasi, data di masukan ke dalam data warehouse yang disebut dengan proses loading.

B. Alur Proses Perancangan Implementasi



Gambar 3 : Alur Perancangan Sistem

Step 1

1. Menganalisis Kebutuhan Data Warehouse dengan Borang BAN-PT

Dalam penelitian ini Borang BAN-PT digunakan sebagai informasi untuk dijadikan acuan kebutuhan yang dibutuhkan untuk membangun data warehouse Pangkalan Data Perguruan Tinggi. Borang BAN-PT memiliki tujuh buah standar untuk Pangkalan Data Perguruan Tinggi, sehingga akan di analisis mengenai bagian mana yang menjadi skema dimensional dan bagian mana yang menjadi skema relasional.

2. Menentukan Tabel Dimensional

Untuk menentukan tabel dimensional dilakukan analisis seperti yang telah dijelaskan sebelumnya. Dari hasil analisis pertanyaan Borang BAN-PT ditentukan delapan tabel fakta dan 13 tabel dimensi. Skema tabel dimensional yang dilakukan pada penelitian ini adalah skema galaxy. Karena satu tabel dimensi dapat digunakan untuk lebih dari satu tabel fakta.

3. Menganalisis Data Source
   Setelah menentukan tabel dimensional dilakukan analisis terhadap data source. Analisis ini dilakukan untuk mencari data apa saja yang dibutuhkan pada datawarehouse. Data source diambil dari sistem informasi Telkom University dan dari localhost.
4. Membuat Tabel Dimensional Pada Data Warehouse
   Setelah ditentukan bagian mana yang menjadi dimensional, kemudian dibuat skema dimensional data warehouse. Tabel dimensional dibuat pada DBMS oracle.

Step 2

1. Membuat Konseptual Model dan Logical Model
   Sebelum membuat logical model, terlebih dahulu dibuat konseptual model. Pada saat membuat konseptual model analisis terlebih dahulu data source yang dibutuhkan untuk memenuhi kebutuhan data warehouse. Pada saat membuat konseptual model kita juga menganalisis transformasi yang dibutuhkan untuk memenuhi kebutuhan data warehouse. Dari konseptual model yang telah dibuat langkah selanjutnya adalah mengkonversi konseptual model dari tabel dimensi

Step 3

Pada langkah ini merupakan penjelasan dari proses ETL dari mulai ekstraksi data ke dalam *tools oracle warehouse builder* sampai dengan melakukan *loading* data ke dalam target (*data warehouse*).

1. Melakukan Ekstraksi Data
   Data diambil dari *souce* dan di ekstrak ke dalam *oracle warehouse builder* dalam bentuk tabel. Seperti gambar-gambar yang tertera diatas, data di ekstraksi dari data source kemudian dalam bentuk table di simpan di dalam *oracle warehouse builder*.
2. Membuat Fungsi Transformasi Data
   Fungsi yang digunakan untuk melakukan transformasi dibuat di dalam *oracle warehouse builder*. Terdapat 16 fungsi yang digunakan untuk transformasi. Enam fungsi diantaranya merupakan fungsi yang telah tersedia pada oracle warehouse builder.
3. Melakukan Pengujian Fungsi Transformasi Data
   Fungsi yang telah dibuat untuk melakukan transformasi diuji apakah fungsi tersebut dapat digunakan untuk melakukan transformasi. Pertama, fungsi yang telah dibuat di validasi terlebih dahulu apakah terdapat *error* atau tidak. Apabila tidak terdapat *error* fungsi tersebut di gunakan di dalam mapping agar ETL dapat langsung di jalankan.
4. Membuat Mapping pada Oracle Warehouse Builder
   Setelah fungsi selesai diuji kemudian fungsi tersebut di aplikasikan pada mapping *Oracle Warehouse Builder*. Proses ETL dilakukan pada *Mapping* di *Oracle Warehouse Builder*. *Mapping* merupakan implementasi dari Konseptual dan Logical Model. Pada *Mapping* proses ETL dari mengekstraksi data, mentransformasi, sampai melakukan loading data ke data warehouse dilakukan.
5. Melakukan Transformasi dan *Loading*
   Setelah selesai membuat *mapping* pada *oracle warehouse builder*, *mapping* yang telah dibuat di validasi. Apabila sudah tidak terdapat kesalahan mapping tersebut dapat di *deploy* untuk memastikan mapping yang kita buat bisa dijalankan atau tidak, serta memastikan *service* pada *oracle warehouse builder* berjalan atau tidak. Setelah selesai melakukan *deploy mapping* dapat di *run*. Pada proses *run* itulah transformasi dan *loading* dilakukan.

Step 4

Pada langkah ini adalah langkah terakhir untuk membuat analisis terhadap proses yang telah dilakukan dan membuat *logical data map*

1. Membuat Logical Data Map
   Logical data map adalah alur yang digunakan pada proses ETL, dimana data tersebut didapat dan bagaimana cara mentransformasikan nya. *Logical data map* dibuat karena pada saat implementasi bisa jadi terputus apabila tidak dirancang secara baik. *Logical data map* menjelaskan dari mana data yang di dapat pada target beserta bagaimana cara mengisi data tersebut.
2. Membuat Analisis validasi data dan performansi proses
   Pada tahap ini yaitu melakukan analisis validasi data dan melihat performansi proses yang telah dilakukan.

C. Tujuan dan Skenario Pengujian

Pada bagian ini akan dipaparkan mengenai tujuan dan skenario pengujian yang dilakukan pada penelitian ini.

1. Tujuan Pengujian
   Pengujian pada penelitian ini bertujuan untuk menentukan kelayakan dari proses yang telah di rancang sebelumnya adapun pengujian yang dilakukan fokus terhadap dua hal yaitu validitas data hasil proses dan performansi proses
2. Skenario Pengujian
   Pengujian yang dilakukan pada penelitian ini akan difokuskan kepada dua hal seperti yang telah dipaparkan sebelumnya. Adapun penjelasan dari skenario pengujian sebagai berikut :
   1. Pengujian validitas data mapping sebelum dilakukan transisi graf dan setelah dilakukan transisi graf: pada pengujian ini akan dilakukan validasi terhadap mapping yang telah dirancang pada oracle warehouse builder. Pada tahap ini adalah tahap dasar untuk menentukan apakah rancangan logical model adalah rancangan yang valid dan equivalen untuk dapat menentukan ke pengujian selanjutnya.
   2. Pengujian performansi proses: pada tahap ini akan di analisis performansi dari proses ETL tersebut. Performansi yang dilihat adalah pada segi waktu, cost dan kapasitas memori yang dgunakan pada saat proses tersebut dilakukan dan dengan melihat apakah baris yang terpilih pada *data source* terinput semua ke dalam *datawarehouse*.
   3. Melakukan perbandingan antara desain satu dengan yang lainnya pada saat melakukan ETL pada tabel dimensi. Dari dua desain atau lebih yang dibuat dilakukan perbandingan desain mana yang lebih optimal. Desain yang dibandingkan adalah

IV. PENGUJIAN DAN ANALISIS

A) Pengujian Validasi Data

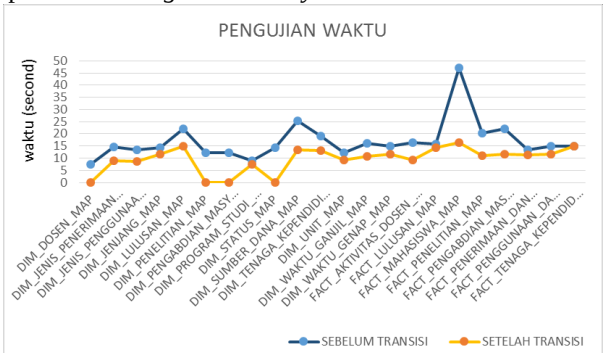
Setelah dilakukan pengujian validasi data pada mapping yang telah dibuat sebelumnya, langkah selanjutnya adalah melakukan pengujian validasi data untuk mapping yang pada logical modelnya telah dilakukan transisi. Dari 22 *logical model* yang telah dibuat, terdapat 15 *logical model* yang dapat dilakukan transisi.

Dari hasil pengujian validasi data yang dilakukan semua data yang telah dilakukan ekstraksi dan

dilakukan transformasi masuk ke dalam proses *loading*. Pada *mapping* dengan logical model sebelum dan dilakukan transisi dan setelah dilakukan transisi merupakan skema yang *equivalen*, hal ini dibuktikan oleh hasil dari ETL yang dilakukan sebelum skema dilakukan transisi dan setelahnya menghasilkan hasil yang sama

B) *Pengujian Performansi ETL Berdasarkan Waktu*

Setelah dilakukan implementasi ETL pada *oracle warehouse builder* dan diuji berdasarkan waktu, terdapat 15 *mapping* yang dapat dilakukan transisi graf pada bentuk *logical modelnya*.



Gambar 4 : Grafik hasil pengujian ETL berdasarkan waktu

gambar 4 dari pengujian waktu ke-1 garis biru merupakan garis waktu sebelum dilakukan transisi dan garis kuning adalah garis setelah dilakukan transisi. Pada garis kuning terdapat titik-titik pada 0.

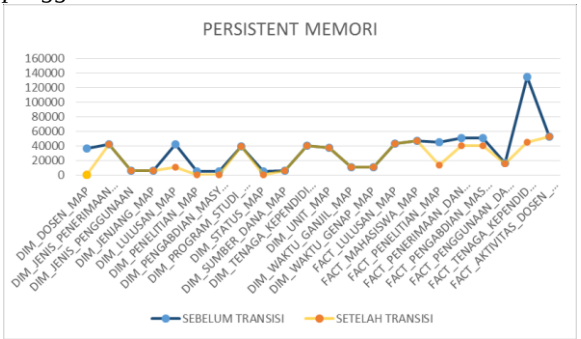
C) *Pengujian Performansi ETL Berdasarkan Cost Proses*

Nilai *cost* didapatkan dari generalisasi *statement* pada saat mengeksekusi *mapping* yang telah dirancang sebelumnya. Berikut adalah hasil performansi ETL berdasarkan *cost* setelah dilakukannya transisi graf

Gambar 5 : Grafik hasil pengujian ETL berdasarkan cost

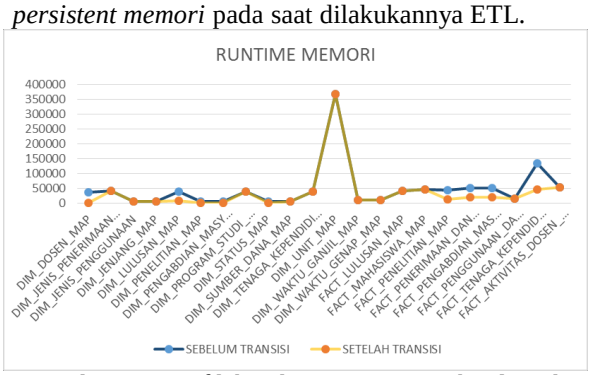
Gambar 5 adalah grafik yang menggambarkan perbandingan pada pengujian berdasarkan *cost*.

D) *Pengujian Performansi proses berdasarkan penggunaan memori*



Gambar 6: Grafik hasil pengujian ETL berdasarkan persistent memori

Gambar 6 adalah grafik yang menunjukan penggunaan *persistent memori* pada saat dilakukannya ETL.



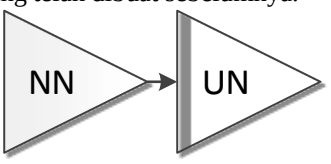
Gambar 7 : Grafik hasil pengujian ETL berdasarkan runtime memori

Gambar 7 adalah grafik yang menunjukan penggunaan memori pada saat melakukan ETL. Apabila dilihat dengan gambar 4-3 grafik menunjukan gambar yang serupa, hal ini dikarenakan *persistent memori* dan *runtime memori* berbanding lurus, penggunaan *runtime memori* dan *persistent memori* tidak memiliki perbedaan yang mencolok. Hal ini dikarenakan *persistent memori* bergantung pada banyaknya kolom yang terdapat pada *query* sedangkan *runtime memory* bergantung pada perintah eksekusinya. Semakin banyak kolom yang dieksekusi pada suatu *statement persistent memori* akan menggunakan memori yang lebih besar. Untuk pengaruh dari *runtime memori* adalah *statement* semakin kompleks suatu *statement* maka memori yang digunakan semakin besar

E) *Analisis Perubahan Aktivitas*

Pada pengujian yang telah dilakukan sebelumnya yaitu pengujian berdasarkan waktu, berdasarkan cost (dalam *bytes*) dan berdasarkan penggunaan memori menunjukan adanya transisi graf pada desain *logical model* menimbulkan hasil yang lebih optimal dari desain sebelumnya. Pada pengujian yang telah dilakukan sebelumnya terlihat waktu yang digunakan untuk melakukan proses ETL membutuhkan waktu yang lebih sedikit apabila dilakukan transisi graf.

Adanya perubahan waktu pada saat dilakukan transisi graf adalah pengaruh dari aktivitas pada proses ETL. Aktivitas tersebut terdiri dari aktivitas transformasi yang telah dipaparkan pada BAB II. Cepat atau lambatnya proses ETL dipengaruhi dari *Query Processing* yang dimana semakin banyak *record* yang di akses maka akan semakin lama pula waktu prosesnya. Hal ini terjadi pada desain proses ETL yang telah dibuat sebelumnya.



Gambar 8 : Gambar urutan aktivitas

Pada gambar 8 merupakan gambar contoh urutan aktivitas sebelum dilakukan transisi graf. Pada aktivitas tersebut aktivitas NN (*Not Null*) adalah aktivitas untuk menghilangkan nilai *null* dan UN(*Unique Value*) adalah aktivitas untuk menghilangkan nilai *redundan*. Aktivitas

tersebut akan lebih cepat apabila dilakukan transisi graf SWAP(NN,UN) yaitu dengan mengubah urutan aktivitas dengan mendahulukan penghilangan nilai *redundan*, hal ini dikarenakan apabila nilai redundan dihilangkan terlebih dahulu akan mempercepat proses ETL. seperti yang dikatakan sebelumnya, semakin banyak *record* pada suatu proses maka proses tersebut akan semakin lama dan *cost* yang di dapatkan akan semakin besar, dengan mengubah urutan aktivitas maka waktu yang digunakan akan lebih cepat dan penggunaan memori lebih optimal.

Perubahan urutan aktivitas harus disesuaikan dengan aktivitas yang lain agar tidak merubah bentuk dari desain *logical model*

F) Analisis Hasil Pengujian

Pada penelitian ini difokuskan kepada memodelkan aktivitas ETL sebagai *state space problem* dengan menjadikan node sebagai aktivitas pada ETL tersebut. Karena merepresentasikan *node* sebagai aktivitas dari ETL maka untuk mendapatkan hasil yang optimal, dilakukan transisi graf pada *logical model* yang telah dibuat sebelumnya untuk implementasi ETL. Sebuah *logical model* yang dibuat harus *equivalen*. *Logical model* dikatakan *equivalen* apabila memiliki *output* yang sama dan kedua skema untuk penyebaran data dari sumber ke target *identik*. Oleh karena itu pada pengujian yang terdapat pada subab 4.2 untuk memastikan skema tersebut *equivalen*. Dari hasil pengujian pada subab 4.2 telah didapatkan hasil skema *equivalen*, untuk itu dapat dilakukan pengujian yang lebih lanjut.

Sebuah transisi graf pada *state space problem* dapat dilakukan apabila memiliki *logical model* yang cukup kompleks. Sebuah *logical model* dikatakan kompleks apabila memiliki lebih dari satu aktivitas yang berurutan untuk menuju suatu tujuan. Misalnya untuk memenuhi suatu kebutuhan pada satu atau lebih *attribute* pada tabel fakta dibutuhkan lebih dari suatu proses.

Pada saat melakukan transisi graf, terdapat perubahan pada urutan eksekusi proses ETL. Untuk melakukan transisi graf hal yang pertama dilakukan adalah memeriksa input dan output dari setiap state. Kemudian dari state-state yang telah di rancang, cari alternatif urutan yang memiliki skema equivalen. Setelah skema ekuivalen baru dapat dilakukan transisi graf. Untuk mengetahui hasil graf lebih optimal yaitu melakukan running dari implementasi yang telah dirancang. Dari hasil pengujian yang telah dilakukan pada subab 4.3, subab 4.4, dan subab 4.5 dilakukannya transisi graf pada *state space problem* dapat mengurangi waktu eksekusi dan penggunaan memori pada saat eksekusi 33,33% sampai dengan 94,73% dari keseluruhan *mapping* yang dilakukan transisi graf. Hal ini disebabkan oleh jumlah record yang di eksekusi lebih sedikit pada mapping yang telah dilakukan transisi graf. Semakin sedikit rows yang di akses maka semakin optimal prosesnya. Pada pengujian yang dilakukan, saat dilakukan transisi graf perbedaan tampak terlihat pada waktu eksekusi, setelah dilakukan transisi graf waktu eksekusi dari proses ETL banyak berkurang. Sedangkan pada cost dan penggunaan memori cenderung sama.

V. KESIMPULAN DAN SARAN

A. Kesimpulan

Berdasarkan penelitian yang telah dilakukan di atas dapat ditarik kesimpulan sebagai berikut :

1. Metode *State-space problem* dapat di implementasikan dalam proses ETL (*Ekstract, Transform, dan Load*) Pangkalan Data Perguruan Tinggi.
2. Pada proses ETL terdapat beberapa ketentuan untuk kebutuhan datawarehouse salah satunya adalah menghilangkan nilai *null* sehingga pada datawarehouse tidak terdapat data yang bernilai *null*.
3. State space problem pada implementasi ini menerapkan proses transisi graf yang dibuat dari *logical model*. *Logical model* setelah dilakukan transisi graf dikatakan valid apabila *logical model equivalen*. Sebuah *logical model* dikatakan equivalen apabila menghasilkan data yang sama. Setelah melakukan pengujian, terbukti mapping sebelum dilakukan transisi dan setelah dilakukan transisi merupakan mapping yang equivalen.
4. Setelah dilakukan transisi graf, hasil performansi dari *logical model* setelah dilakukan transisi lebih optimal mencapai 46,67-53,33%

B. Saran

1. Perlu adanya penyesuaian fungsi transformasi pada lingkungan yang berbeda.
2. Penelitian ini dapat dikembangkan kedalam optimasi performansi dari ETL berdasarkan penyimpanan dengan menggunakan metode yang serupa.
3. Sebaiknya mempertimbangkan penggunaan memori database untuk lebih besar dari 60GB apabila memiliki proses yang kompleks dan jumlah data yang besar.
4. Penelitian ini bisa dikembangkan dengan menggunakan database dari *host* yang berbeda dengan berbagai jenis format file.

VI. DAFTAR PUSTAKA

[1] Imhof, C. (2014, July 18). Retrieved from The Corporate Information Factory: http://www.information-management.com/issues/19991201/1667-1.html

[2] Juman, K. K. (2013, September 17). Corporate Sistem Informasi. Retrieved from Extract, Transform, Load: http://kundang.weblog.esaunggul.ac.id/2013/09/17/extract-transform-loading/

[3] Kimball, R., & Casetra, J. (2004). The Data Warehouse ETL Toolkit. New York: John Wiley and Sons.

[4] Kimlball, R., Reeves, L., Ross, M., & Thornthwaite, W. (1998). The Data Warehouse Lifecycle Toolkit. New York: John Wiley an Sons.

[5] Paulraj, P. (2001). Data Warehouse Fundamentals: a Comprehensive . New York: John Wiley & Sons, Inc.

[6] Pole, D., & Mackworth, A. (2014, September 28). Fondation of Computational Agents. Retrieved from Artificial Intelegence: http://artint.info/html/ArtInt_48.html

[7] Rainardi, V. (2008). Building a Data Warehouse with Examples in SQL Server. New York: Springer-Verlag.

[8] Simitsis, A. (2004). Modeling and Optimizing of Extraction-Transformation-Loading (ETL) Processes in Data Warehouse. Athena: Ph.D. Electrical and Computer Engineering N.T.U.A.

[9] State Space. (2014, September 28). Retrieved from http://www.cs.miami.edu/~geoff/Courses/COMP6210-10M/Content/StateSpaceSearch.shtml

[10] Vasilliadis, P., Simitsis, A., & Skiadopoulus, S. (2002). Conceptual Modeling for ETL Processes.
[11] Velicanu, M., & (TEOHARI), L. C. (2013). Enhancing ETL Performance with Warehouse Builder.