***LAPORAN PENELITIAN***

***Data Warehouse Design With ETL Method (Extract, Transform, And Load) for Company Information Centre***

**Program Studi : Teknik Informatika**

**Jenjang Pendidikan : Strata 1**

****

Diajukan Oleh :

**Wulan Staufana**

**16101152630051**

**Dosen Pengampu**

**Rini Sovia , M.Kom / 1005047601**

**Randy Permana, M.Kom / 1012128701**

**PROGRAM STUDI TEKNIK INFORMATIKA**

**FAKULTAS ILMU KOMPUTER**

**UNIVERSITAS PUTRA INDONESIA “YPTK”**

**PADANG**

**2020**

# ABSTRAK

Data Warehouse adalah penggunaan teknologi untuk menganalisis, mengekstrak dan mengevaluasi data menjadi informasi yang menghasilkan pengetahuan dalam bentuk analisis untuk memberikan saran dalam proses pengambilan keputusan. Merancang Data Warehouse menggunakan proses ETL (Extract, Transformation and Load) berfungsi sebagai pengumpulan data dari berbagai sumber data ke dalam banyak kumpulan data terintegrasi. Dengan menggunakan skema snowflake untuk perancangan data warehouse membuat data siap untuk dianalisa pada Data Warehouse. Hasil dari penelitian ini adalah penerapan Data Warehouse yang digunakan untuk mendukung kemajuan pengambilan keputusan perusahaan menjadi lebih mudah dan memiliki keputusan yang baik karena berasal dari Data Warehouse

**Kata Kunci : Gudang Data, ETL, Skema Kepingan Salju, Konsultan, OLAP**

DAFTAR ISI

[BAB I PENDAHULUAN 5](#_Toc44279053)

[1.1. Latar Belakang Masalah 5](#_Toc44279054)

[1.2. Perumusan Masalah 7](#_Toc44279055)

[1.3. Hipotesa 7](#_Toc44279056)

[1.4. Batasan Masalah 8](#_Toc44279057)

[1.5. Tujuan Penelitian 8](#_Toc44279058)

[1.6. Manfaat Penelitian 9](#_Toc44279059)

[1.7. Deskripsi Objek Penelitian 9](#_Toc44279060)

[BAB II LANDASAN TEORI 1](#_Toc44279064)0

[2.1. Rekayasa Perangkat Lunak (RPL) 1](#_Toc44279065)0

[2.2 Software Development Life Cycle 1](#_Toc44279071)5

[*2.3.* *Flowchart* 2](#_Toc44279074)3

[2.4. *Unified Modelling language* 2](#_Toc44279081)8

[2.5. *Database* 4](#_Toc44279084)0

[*2.6.* *Sistem Manajemen Basis Data* 4](#_Toc44279085)0

[2.7. Entiti Relationship Model 4](#_Toc44279086)1

[2.8. Entity Relationship Diagram 4](#_Toc44279087)6

[2.9. Datawarehouse 4](#_Toc44279088)9

[2.10. ETL 5](#_Toc44279089)9

[BAB III METODOLOGI PENELITIAN 8](#_Toc44279090)1

[3.1 Kerangka Penelitian 8](#_Toc44279091)1

[3.2 Tahapan Penelitian 8](#_Toc44279092)1

[3.2.1 Penelitian Pendahuluan 32](#_Toc44279093)

[3.2.2 Pengumpulan Data 32](#_Toc44279094)

[3.2.3 Penganalisa Masalah 35](#_Toc44279095)

[3.2.4 Perancangan 36](#_Toc44279096)

[3.2.5 Implementasi 38](#_Toc44279097)

[3.2.6 Pengujian 39](#_Toc44279098)

[3.2.7 Evaluasi 39](#_Toc44279099)

[BAB IV ANALISA DAN PERANCANGAN .](#_Toc44279090)144

[**DAFTAR PUSTAKA 9**](#_Toc44279128)**2**

# BAB I

**PENDAHULUAN**

* 1. **Latar Belakang Masalah**

*Teknologi* informasi di dunia pembangunan khususnya bagian konsultasi merupakan pendukung terbesar dalam mencapai sebuah keunggulan untuk memajukan kesejahteraan perusahaan. Teknologi sangat berperan penting dalam kehidupan manusia pada era perkembangan zaman sekarang ini. Salah satu hal yang terpenting dalam bidang jasa konsultasi yang menyebabkan kebutuhan teknologi sangat dibutuhkan yaitu teknologi informasi, dimana seluruh data serta informasi sangat penting yang menyebabkan kebutuhan data semakin meningkat dari tahun ke tahun. Data transaksi merupakan data yang sangat penting bagi suatu perusahaan yang melakukan pencatatan untuk merekam semua transaksi yang terjadi. Selama ini seperti pembukuan fisik, dan pembukuan digital yang disimpan dalam berbagai bentuk format dengan penyimpanan yang relational.

CV. Restu Graha Cipta merupakan sebuah perusahaan yang bergerak di bidang konsultasi yang mana pada perusahaan menerapkan sistem berbasis komputer pada transaksinya, data-data yang disimpan akan semakin banyak dan semakin kompleks untuk dilakukan pengolahan. Data yang disimpan tersebut mengakibatkan kerumitan dalam pengolahan yang disebabkan penambahan data dari kegiatan yang dilakukan setiap hari oleh perusahaan dan data yang telah lama akan terabaikan dengan adanya pemasukan data yang baru.

*Data warehouse* adalah tempat data yang menyimpan data yang berskala luas. Menurut Kimball dan Ross (2004), *data warehouse* adalah sebuah sistem yang mengekstrak, membersihkan, menyesuaikan dan menyampaikan sumber data kedalam sebuah tempat penyimpanan data dimensional dan kemudian mendukung dan mengimplementasi *querry* dan *analysis* untuk tujuan pengambilan keputusan. *Data warehouse* menampung data secara keseluruhan dan menyajikan kembali data tersebut untuk keperluan pelaporan dan *analysis* data yang bertujuan untuk menunjang pengambilan keputusan yang strategis berdasarkan pada analisis data. Pemetaan data perusahaan ini bertujuan untuk dapat menggambarkan kondisi informasi mengenai perusahaan dari berbagai dimensi data yang didapat. *Data warehouse* akan dijadikan sebagai dasar dalam pengambilan keputusan strategis pada perusahaan.

Berdasarkan penelitian di CV. Restu Graha Cipta dapat disimpulkan bahwa *data warehouse* belum tersedia dan belum untuk melayani *strategic* dari proses pengambilan keputusan di perusahaan tersebut. Data yang terdapat dalam perusahaan merupakan informasi yang sangat berharga yang mempengaruhi kelangsungan dari perusahaan. Tetapi perusahaan belum memiliki pengaksesan informasi yang mudah diakses, disamping itu kurangnya *privacy* dan keamanan informasi yang dimiliki oleh perusahaan. Hal ini memicu kurang berkembang perusahaan dalam mendukung perubahan-perubahan kedepannya baik perubahan yang terjadi secara *internal*, *eksternal*, maupun kedua-duanya yang berskala kecil maupun berskala besar.

CV. Restu Graha Cipta merupakan objek penelitian yang dipilih, hal ini dikarenakan pada perusahaan tersebut data belum dikelola secara maksimal. Pada perusahaan data yang disimpan dalam bentuk format *png*, *jpg*, *xlx*, *doc* dan *pdf* Sehingga apabila mereka mengolah data dari tahun sebelumnya mereka akan mengalami kerumitan untuk mencari data-data sebelumnya. Dan data disimpan juga tidak memiliki manfaat yang maksimal sehingga resiko dalam pengambilan keputusan dan kualitas dan konsistensi data di perhitungkan. Hal ini dikarenakan perusahaan saat ini belum mengenal metode data untuk penggalian informasi berupa metode ETL (*extract transform* and *load*) yang berada pada *data warehouse*.

Diharapkan dari penelitian yang berjudul **“Perancangan *Data warehouse* Dengan Menggunakan Metode *Extract, Transform,* and *Load* di CV. Restu Graha Cipta Padang”** ini dapat membantu pihak perusahaan dalam mengambil keputusan serta melakukan tindakan yang tepat dari hasil akhir pengolahan data.

* 1. **Perumusan Masalah**

Berdasarkan latar belakang masalah diatas, maka adapun perumusan masalah dari penulisan ini adalah :

1. Bagaimana proses memilih dan menyeleksi (*ekstraksi*)data yang berasal dari berbagai sumber data yang telah ditentukan ?
2. Bagaimana proses untuk mengubah data menjadi bentuk yang diinginkan(*transform*) untuk kebutuhan *analysis* data ?
3. Bagaimana proses yang merupakan penyajian(*load*)data yang telah di transformasikan dapat menyimpan data dan menghasilkan berbagai macam output ?
4. Bagaimanadapat membantu strategi kedepannya dalam pengambilan keputusan pada perusahaan dari data-data informasi yang didapatkan seperti data fasilitas, peralatan atau perlangkapan pendukung, data tenaga ahli/personalia, data pengalaman perusahaan (proyek) dari output kepada pimpinan?
   1. **Hipotesa**

Berdasarkan perumusan masalah tersebut, maka hipotesa yang didapat yaitu :

1. Dengan *data warehouse* ini, dapat melakukan extraction terhadap data dari berbagai sumber.
2. Dengan *data warehouse* ini, diharapkan dapat melakukan transformation data atau mengubah data kedalam format yang diinginkan.
3. Dengan *data warehouse* ini, diharapkan dapat melakukan loading atau penyajian data sebagai informasi untuk meneruskan data ke dalam *data warehouse* untuk di simpan kepada pengguna akhir.
4. Diharapkan dengan dirancangnya *data warehouse* ini, dapat membantu strategi dalam kelancaran dalam pengambilan keputusan pada perusahaan dari output yang dihasilkan.

* 1. **Batasan Masalah**

Agar penulisan ini sesuai dengan yang diharapkan dan penelitian ini tidak melewati batas dari tujuan dan lebih terarah dalam proses pengumpulan data dan informasi maka dibuatlah batasan suatu masalah yaitu :

1. Merancang *data warehouse* yang mencangkup data fasilitas, tenaga ahli atau personalia, dan pengalaman perusahaan atau proyek pada CV. Restu Graha Cipta Padang.
2. Informasi yang didapat bersifat kuantitatif.
3. Informasi yang didapat tidak membahas data mining.
4. Data yang digunakan adalah data *historys* yang terdiri dari 5 tahun.
   1. **Tujuan Penelitian**

Adapun tujuan yang dimiliki penelitian dalam perancangan *data warehouse* ini adalah:

1. Memudahkan pihak perusahaan dalam pengambilan keputusan serta tindakan terhadap hasil akhir berupa informasi karena pengolahan data yang dilakukan.
2. Memudahkan pihak perusahaan dalam melihat pergerakkan perusahaan selama beberapa tahun secara keseluruhan dari data yang telah ada .
3. Memudahkan dalam pencarian data-data yang telah lama.
4. Untuk memantau perkembangan perusahaan.
5. Untuk meningkatkan kinerja yang lebih efisen dan berkualitas bagi perusahaan.
6. Untuk memprediksi suatu keadaan perusahaan dimasa yang akan datang
7. Membantu perusahaan untuk membuat strategi terhadap tantangan-tantangan yang ada pada perusahaan.
   1. **Manfaat Penelitian**

Adapun pada penelitian ini terdapat manfaat yang dimiliki yaitu:

1. Mempermudah dalam melihat data-data secara keseluruhan dengan mudah dan cepat.
2. Membantu dalam pengambilan keputusan yang tepat dari hasil data yang telah diolah.
3. Memudahkan untuk dapat memantau perkembangan perusahaan.
4. Meningktakan kinerja perusahaan untuk kedepannya dan secara keseluruhan.

# BAB II

**TINJAUAN PUSTAKA**

## 2.1 Rekayasa Perangkat Lunak

## 2.1.1 Defenisi Rekayasa Perangkat Lunak

Pada suatu perancangan aplikasi hal yang paling dominan dilakukan adalah memodelkan kebutuhan pemakai. Terdapat banyak cara memodelkan aplikasi seperti halnya banyak cara yang digunakan oleh seorang arsitek untuk membangun sebuah bangunan. Pada umumnya pemodelan merupakan gabungan antara perangkat lunak dan perangkat keras yang digunakan. Perangkat lunak (*software*) program komputer terasosiasi dengan dokumentasi perangkat lunak yaitu dokumentasi kebutuhan, model desain, dan *user manual*. Program komputer apabila tanpa terasosiasi dengan dokumentasinya hal tersebut belum disebut sebagai perangkat lunak (*software*). Sistem perangkat lunak merupakan sistem yang memiliki komponen yang memiliki hubungan antara satu dengan yang lain untuk memenuhi kebutuhan pelanggan (*customer*) (Rosa and M.Shalahuddin, 2018:2)

Menurut (Rosa and M.Shalahuddin, 2018:2) ada 3 karakter yang terdapat pada perangkat lunak yaitu:

1. Perangkat lunak dibangun dari rekayasa (*software engineering*) tidak di produksi secara manufaktur dan pabrikan.
2. Perangkat lunak tidak pernah usang (*wear out*) sebab kecacatan pada perangkat lunak yang dimiliki dapat diperbaiki.
3. Barang yang diproduksi pabrik biasanya komponen baru pada barang tersebut terus di produksi, tetapi perangkat lunak akan terus diperbaiki seiring dengan bertambahnya kebutuhan yang diperlukan.

Rekayasa perangkat lunak disebut sebagai teknik penerapan ilmu dan teknologi untuk menyelesaikan permasalahan manusia yang dapat selesai lewat pengetahuan, matematika, dan pengalaman secara praktis yang diterapkan untuk mendesain suatu objek atau proses yang berguna (Simarmata, 2010:10). Rekayasa perangkat lunak yaitu pembangunan yang menggunakan prinsip serta konsep rekayasa yang bertujuan untuk menghasilkan perangkat lunak bernilai ekonomi dan dipercaya serta bekerja secara efisien menggunakan mesin. Rekayasa perangkat lunak sangat dibutuhkan karena perangkat lunak yang dibuat tidak menjadi perangkat yang tidak berguna (Rosa and M.Shalahuddin, 2018:4).

Pada tahun 2004, rekayasa perangkat lunak secara umum digunakan dalam 3 istilah yang memiliki pengertian (Simarmata, 2010:11) sebagai berikut :

1. Sebagai istilah umum yang digunakan dalam berbagai kegiatan yang dulunya diberi nama yaitu pemograman atau *analysis system*.
2. Merupakan istilah luas yaitu sebagai analisis teknik dari semua aspek – aspek praktis yang bertentangan dengan teori pemograman komputer.
3. Sebagai istilah demi mewujudkan *advokasi* suatu pendekatan spesifik ke programan komputer, suatu hal yang mendesak diperlukan sebagai profesi rekayasa dari pada sebuah seni atau kerajinan, dan *advokasi* dari konfikasi praktis yang disarankan dalam bentuk metodologi rekayasa perangkat lunak.

## 2.1.2 Kriteria Perangkat Lunak

Rekayasa perangkat lunak lebih fokus kearah bagaimana membuat perangkat lunak untuk dapat memenuhi kriteria sebagai berikut (Rosa and M.Shalahuddin, 2018:4):

1. Dapat dipelihara secara terus menerus setelah perangkat lunak selesai dibuat seiring berkembangnya pada teknologi dan lingkungan (*maintainability*).
2. Untuk dapat diandalkan sebagai proses bisnis yang dijalankan dan perubahan yang terjadi (*dependability* dan *robust*).
3. Efisien dari segi yang terdapat dari sumber daya dan pengguna.
4. Kemampuan dimana dapat dipakai sesuai dengan kebutuhan (*usability*).

Dari kriteria yang terdapat diatas, perangkat lunak yang baik yaitu perangkat lunak yang dapat memenuhi kebutuhan pelanggan. *Costumer* atau *user* (pemakai perangkat lunak) atau berorientasi pada pelanggan atau pemakai perangkat lunak, bukan berorientasi pada pembuat atau pengembang perangkat lunak.

## 2.1.3 Tantangan Proses Rekayasa Perangkat Lunak

Menurut (Rosa and M.Shalahuddin, 2018:7) tantangan yang dihadapi pada proses rekayasa perangkat lunak yaitu :

1. Tantangan warisan dari perangkat lunak yang dikembangkan selama bertahun-tahun oleh orang-orang berbeda, hal itu menyebabkan ketidakpahaman atau perubahan tujuan pembuatan perangkat lunak.
2. Tantangan *heterogenitas* yaitu perangkat lunak harus beradaptasi dengan teknologi yang semakin berkembang dengan luasnya lingkungan distribusi perangkat lunak.
3. Tantangan pengiriman dimana perangkat lunak dengan skala besar dan kompleks sekalipun dapat sampai ke tangan pelanggan (*costume*r) atau *user* secara cepat dan tetap terjaga kualitasnya
4. Tantangan utama dalam pengembangan perangkat lunak terletak dalam pembuatan kode baru, tetapi persaingan luar biasa menyebabkan pemaksaan solusi perangkat lunak yang tepat waktu, anggaran dan target yang pas.

Terdapat tantangan utama dalam menyikapi pengembangan perangkat lunak yaitu sebagai berikut :

1. *User power* and *authority*

*User power* and *authority* merupakan pengguna user dan individu yang lebih kuat, berpengalaman, dan selektif.

1. *Market share*

Pesaing akan menjadi lebih teliti, kreatif, dan memiliki banyak ilmu pengetahuan.

1. *Killer applications*

Aplikasi yang digunakan akan lebih bisa *dikustomisasi*, *interaktif*, *dinamis* serta penuh dengan inovasi.

1. *The anytime-anywhere factor*

Sebagai akomodasi aplikasi web dan ruang lingkup tanpa kabel (*wireless*) juga peningkatan beberapa alat portebel.

1. *The return on investment (ROI) factor*

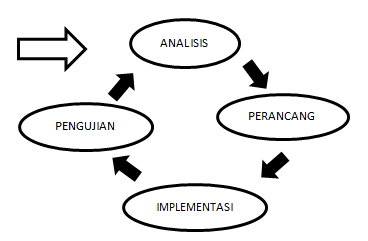
Perangkat lunak sebagai investasi bukan hanya kegiatan teknis. Maka pengevaluasian dilakukan pada nilai yang dihasilkan yang dilihat perbedaan dengan kemampuan yang bisa dikirimkan.

1. *The technology factor*

Berasal berbasis *web* dan *CASE* hingga mengintegrasikan perangkat lunak dan keras kedalam aplikasi serta peralatan multimedia, yang mengalami peningkatan pada tingkat cepat. Hal ini membuat penyelidikan dapat memahami hal yang berguna untuk pengembangan demi menambah keuntungan kompetitif jika digunakan dengan baik.

## 2.1.4 Proses Rekayasa Perangkat Lunak

Proses rekayasa perangkat lunak dapat dilakukan apabila pembangunan perangkat lunak terus berlangsung. Proses–proses dilakukan dalam rekayasa perangkat lunak secara besar hal sebagai berikut :



Sumber : **S, Rosa A dan M. Shalahuddin, 2018**

**Gambar 2.1 Tahapan Umum Rekayasa Perangkat Lunak**

Proses diatas dapat dilakukan berulang-ulang sampai perangkat tersebut dapat memenuhi kebutuhan pelanggan dan *user*. Perangkat lunak (*software process*) sekumpulan aktifitas memiliki tujuan dalam mengembangkan atau mengubah perangkat lunak. Secara umum proes perangkat lunak terdiri atas (Rosa and M.Shalahuddin, 2018:11):

1. Pengumpulan Spesifikasi (*Specification*)

Mengetahui apa yang harus dapat dilakukan oleh sistem perangkat lunak dan batasan yang terdapat pada pengembangan perangkat lunak.

1. Pengembangan *(Development)*

Pengembangan perangkat lunak dalam menghasilkan sistem perangkat lunak.

1. Validasi *(Validation)*

Memeriksa jika perangkat lunak susah dalam memenuhi kebutuhan pelanggan *(customer)*.

1. Evolusi *(Evolution)*

Mengubah perangkat lunak guna memenuhi perubahan kebutuhan pelanggan *(customer)*.

## 2.2 Software Development Life Cycle

## 2.2.1 Definisi Software Development Life Cycle

SDLC (*Software Development Life Cycle)* atau disebut juga sebagai *System Development Life Cycle* merupakan proses mengembangkan atau mengubah suatu sistem perangkat lunak menggunakan model-model dan metodelogi yang digunakan untuk mengembangkan sistem-sistem perangkat lunak sebelumnya (berdasarkan *best practice*) (Rosa and M.Shalahuddin, 2018:26).

SDLC (*Software Development Life Cycle*) telah ada pada tahun 1960-an. Hingga saat ini SDLC (*Software Development Life Cycle*) dapat berguna untuk pengembangan sistem berskala besar secara fungsional untuk konglomerat. Sistem yang didirikan dalam SDLC ini demi mengelola informasi dan aktifitas dari perusahaan yang memiliki potensi terhadap sebuah data yang besar dalam perkembangannya.

## 2.2.2 Tahapan-tahapan Software Development Life Cycle

Menurut (Rosa and M.Shalahuddin, 2018:26) ada beberapa tahapan yang digunakan dalam *Software Development Life Cycle* secara umum sebagai berikut :

1. Inisiasi (*initiation*)

Inisiasi merupakan proses tahapan pertama, pada tahapan ini terjadinya proses dalam melakukan pembuatan proposal terlebih dahulu sebelum membuat proyek perangkat lunak.

1. Pengembangan konsep sistem (*system concept development*)

Pengembangan konsep sistem yaitu menjelaskan ruang lingkup konsep yang terdiri dari dokumen lingkup sistem, analisis manfaat biaya, manajemen rencana, dan pembelajaran kemudahan sistem.

1. Perencanaan (*planning*)

Pengembangan *planning* merupakan pengembangan manajemen proyek dan dokumen perencanaan lainnya. Menyiapkan dasar untuk mendapatkan sumber daya (*resources*) yang diperlukan demi mendapatkan solusi.

1. Analis kebutuhan (*requirement analysis*)

Analisis kebutuhan merupakan kegiatan menganalisis keperluan pengguna sistem perangkat lunak dan mengembangkan keperluan *user* demi membuat kebutuhan dokumen fungsional.

1. Desain (*design*)

Design merupakan proses dalam menstrasformasikan keperluan detail menjadi kebutuhan yang terperinci, dalam dokumen sistem hanya terfokus pada bagian fungsi yang memenuhi fungsi berdasarkan kebutuhan.

1. Pengembangan (*development*)

Mengonversi desain kedalam sistem informasi secara detail termasuk mendapatkan serta melakukan aktifitas instalasi lingkungan sistem yang diperlukan.

1. Integrasi dan pengujian (*integration and test*)

Mendemonstrasikan sistem perangkat lunak yang telah siap memenuhi kebutuhan yang dispesifikasi pada dokumen kebutuhan fungsional. Dari hasil tersbeut menghasilkan laporan analisis pengujian.

1. Implementasi (*implementation*)

Penerapan perangkat lunak pada lingkungan *user*  dan melakukan resolusi dari masalah yang terdeteksi pada fase integrasi dan pengujian.

1. Operasi dan pemeliharaan (*operations and maintenance*)

Menjelaskan pekerjaan dalam pengoperasian dan merawat sistem informasi pada lingkup produksi yang terdiri dari penerapan akhir dan masuk menuju tahapan peninjauan.

1. Disposisi (*disposition*)

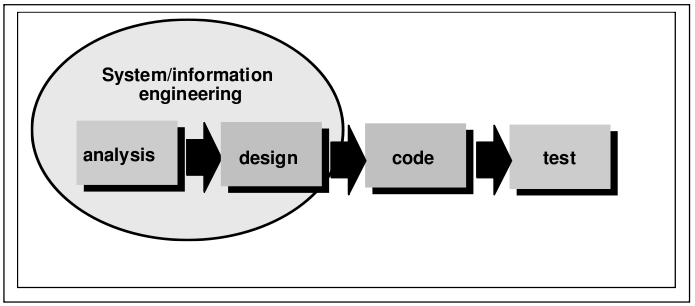
Menjelaskan aktifitas *finish* pengembangan sistem dan mebangun data real sesuai dengan kegiatan *user*.

## 2.2.3 Model Software Development Life Cycle

Menurut (Rosa and M.Shalahuddin, 2018:28) *Software Development Life Cycle* memiliki beberapa model dalam penerapan untuk tahapan prosesnya yaitu :

1. **Model *Waterfall***

Model SDLC air terjun *(waterfall)* disebut juga sebagai model sekuensial liniear (*sequential linear*) atau alur hidup klasik (*classic life cycle)* . Model *waterfall* menyediakan pendekatan alur hidup perangkat lunak yang sekuensial atau terurut dimulai dengan analisis, desain, pengodean, pengujian, dan tahap pendukung (*support).* Berikut gambar model *waterfall* :



Sumber : **S, Rosa A dan M. Shalahuddin 2018**

**Gambar 2.2 Model Waterfall**

Pada tahapan *waterfall* memiliki 5 tahapan dalam prosesnya yang terdiri sebagai berikut ini :

1. Analisis kebutuhan perangkat lunak (*analysis*)

Pada proses ini pengumpulan kebutuhan dapat dilakukan secara intensif dalam menspesifikasi kebutuhan perangkat lunak dimana mudah memahami perangkat lunak yang dibutuhkan oleh *user*. Pada tahap ini sangat diperlukan untuk didokumentasikan.

1. Desain (*design*)

Merupakan proses multi langkah yang fokus terhadap desain pembuatan program perangkat lunak yang terdiri juga dari struktur data, aristektur perangkat lunak, representasi antarmuka, dan prosedur pengodean. Pada desain mentranslasi kebutuhan dari tahap analisis kebutuhan ke representasi desain untuk dapat diimplementasikan menjadi program pada proses selanjutnya.

1. Pembuatan kode program (*code*)

Pada tahap ini program komputer harus sesuai dengan desain yang telah dibuat.

1. Pengujian (*test*)

Pada pengujian lebih fokus terhadap perangkat lunak dari segi *logic* dan fungsional serta memastikan semua bagian telah diuji. Agar mengurangi kesalahan (*error*) dan memastikan *output* yang dihasilkan sesuai yang diinginkan.

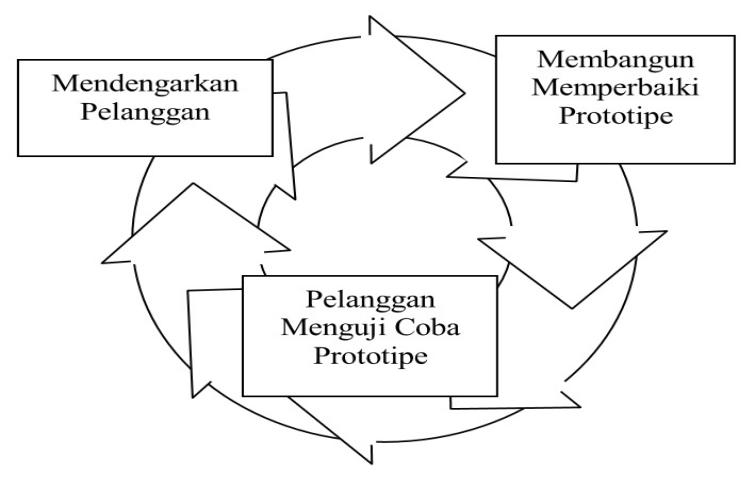
1. Pendukung (*support*) atau pemeliharaan (*maintenance*)

Pada tahap ini tidak terdapat pada proses *waterfall* tetapi tidak menutup kemungkinan pada perangkat lunak mengalami perubahan apabila sudah berada ditangan *user.* Perubahan terjadi akibat kesalahan yang muncul dan tidak terdeteksi pada pengujian atau perangkat lunak harus beradaptasi dengan lingkungan baru. Pada tahap ini dapat mengulangi proses analsis dalam pengembangan tetapi tidak terjadinya pembuatan perangkat lunak baru.

1. **Model Prototipe**

Prototipe merupakan bagian dari produk untuk mengekspresikan logika atau fisik antarmuka *eksternal* yang ditampilkan. Konsumen potensial menggunakan prototipe dan menyediakan masukan sebelum pengembangan skala besar dimulai (Simarmata, 2010:62).

Pada model prototipe dapat digunakan untuk menyambungkan ketidakpahaman pelanggan mengenai hal teknis dan memperjelas spesifikasi kebutuhan yang pelanggan terhadap pengembang perangkat lunak. Sering pelanggan (*customer*) membayangkan dimana kumpulan kebutuhan yang diinginkan namun tidak terspesifikasi secara detail dari segi masukan (*input*), proses, ataupun keluaran (*output*). Pengembang perangkat lunak harus menspesifikasikan kebutuhan secara detail dari segi teknis yang mana pelanggan sering kurang dalam mengenai hal teknis tersebut (Rosa dan Shalahuddin, 2018:31).



Sumber : **S, Rosa A dan M. Shalahuddin, 2018**

**Gambar 2.3 Model Prototipe**

*Mock –Up* suatu hal yang digunakan sebagai model desain yang dilakukan dalam mengajar, demostrasi, evaluasi desain, promosi atau keperluan lain. Sebuah *Mock–Up* disebut juga *prototipe* perangkat lunak untuk menyediakan ataupun mendemostrasikan sebagian besar fungsi sistem perangkat lunak serta memungkinkan pengujian desain sistem perangkat lunak, iterasi terjadi pada pembuatan prototipe harus sesuai dengan keinginan pelanggan (*customer*) atau *user*.

Model prototipe memiliki berbagai macam kelemahan yang ada sebagai berikut :

1. Pelanggan sering mengubah atau menambahkan spesifikasi kebutuhan. Karena iterasi ini menyebabkan banyaknya perubahan atau penambahan spesifikasi kebutuhan pada perangkat lunak, hal ini membuat pengembang banyak mengalah terhadap pelanggan.
2. Para pengembang sering berkompromi dengan pelanggan untuk mendapatkan prototipe dengan waktu yang cepat guna menghasilkan prototipe untuk didemonstrasikan. Dengan hal ini dapat menyebabkan kualitas yang ada pada perangkat lunak akan kurang baik atau menyebabkan iteratif tanpa akhir.

Model prototipe kurang cocok untuk aplikasi dengan skala yang besar sebab dalam membuat prototipe untuk aplikasi skala besar membutuhkan waktu serta tenaga (Rosa dan Shalahuddin, 2018:32)*.*.

1. **Model *Rapid Application Development* (RAD)**

*Rapid Application Development* (RAD) merupakan sebuah model untuk proses pengembangan perangkat lunak yang memiliki sifat *inkremental* teruntuk pada waktu pengerjaan yang pendek. Pada model ini merupakan adaptasi dari model *waterfall* versi kecepatan tinggi yang menggunakan model *waterfall* untuk pengembangan pada tiap komponen yang terdapat pada perangkat lunak (Rosa dan Shalahuddin, 2018:34).



Sumber : **S, Rosa A dan M. Shahaludi, 2018**

**Gambar 2.4 Model RAD**

Pada model *Rapid Application Development (*RAD) memiliki beberapa tahap serta penjelasan sebagai berikut :

1. Permodelan bisnis

Permodelan bisnis yaitu memodelkan fungsi bisnis untuk mengetahui informasi yang berkaitan dengan bisnis, informasi yang dibuat, siapa pembuat informsi bagaimana alur informasi dan prises apa yang terkait pada informasi tersebut.

1. Permodelan data

Permodelan data berupa memodelkan data yang dibutuhkan berdasarkan permodelan bisnis dan mendefinisikan atribut-atribut beserta relasi data yang lainnya.

1. Permodelan proses

Tahap untuk mengimplementasikan fungsi bisnis yang didefinisikan terkait dalam pendefinisikan data.

1. Pembuatan aplikasi

Mengimpelentasikan proses dan data dalam bentuk program.

1. Pengujian dan pergantian

Untuk menguji komponen yang telah dibuat. Apabila sudah diuji maka tim bagian pengembang komponen dapat mengembangkan komponen selanjutnya.

Selain itu menurut (Rosa dan Shalahuddin, 2018: 36) Pada model RAD adanya kelemahan yang dimilikinya pada model tersebut yaitu :

1. Pada pembuatan sistem perangkat lunak dengan skala besar, model RAD memerlukan sumber daya manusia yang cukup besar demi membentuk tim-tim dalam mengembangkan komponen.
2. Apabila tidak adanya persetujuan dalam pengembangan perangkat lunak secara cepat (*rapid*) maka proyek ini akan gagal karena tidak kejelasan dalam mendefinisikan kebutuhan pelanggan (*customer*) atau *user*.
3. Apabila sistem perangkat lunak yang dibuat tidak dapat dibagi menjadi beberapa komponen maka model RAD tidak bisa digunakan dalam membuat sistem perangkat lunak disebabkan banyaknya campur tangan antar kelompok.
4. Pada model ini tidak dapat digunakan pada sistem perangkat lunak yang memiliki risiko teknis yang tinggi, contoh menggunakan teknologi yang belum dikuasai dan belum dikenal banyak.
5. **Model Iteratif**

Pada model iteratif (*iterative model*) menggabungkan proses model *waterfall* dan iteratif pada model *prototipe*. Model *inkramental* dapat menghasilkan versi perangkat lunak yang dapat mengalami penambahan fungsi pada tiap pertambahan inkremen / *increment*.

Sistem/Rekayasa Informasi

Inkremen 1

Poduk Inkremen 1

Uji

Kode

Desain

Analisis

Poduk Inkremen 2

Inkremen 2

Uji

Kode

Desain

Analisis

Uji

Kode

Desain

Analisis

Inkremen 3

Poduk Inkremen 3

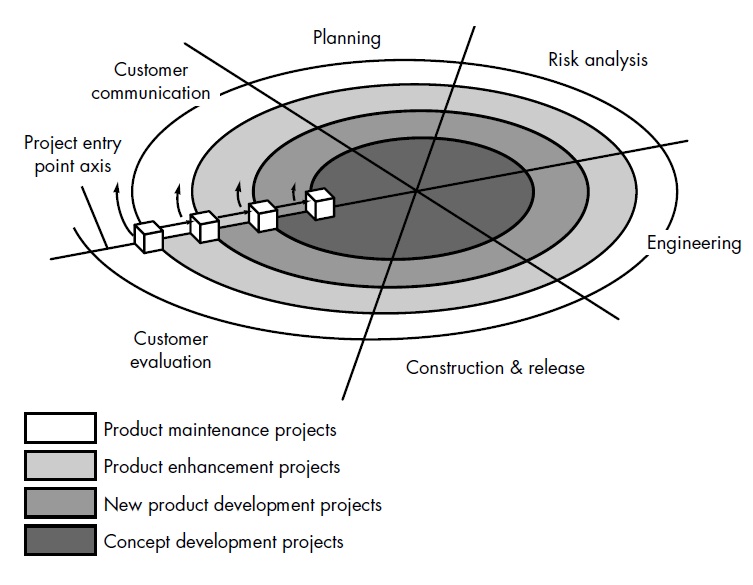
Sumber : **S, Rosa A dan M. Shalahuddin, 2018**

**Gambar 2.5 Model Iteratif**

Model *inkremental* digunakan dalam mengatasi dari kelemahan *waterfall* dimana pada model tersebut tidak mengakomodasi iterasi, dan mengatasi kelemahan dimana memiliki proses yang pendek dan pada setiap iteratif proses tidak terus menghasilkan produk. Model *inkremental* menghasilkan produk atau aplikasi pada tiap proses *inkremental*.

1. **Model Spiral**

Model spiral dikembangkan Boehm (1988) dikembangakan karena pengalaman dengan berbagai perbaikan model *waterfall* yang di implementasikan pada proyek pemerintah, terkhusus pada perangkat lunak besar. Pada model ini memiliki konsep yang mana setiap bagian produk dan tingkatan terdiri dari urutan yang sama pada langkah-langkah serta siklusnya. Model spiral juga menyediakan pengembangan yang cepat dengan memiliki versi yang bertambah fungsinya (*increment*). Iterasi awal menghasilkan sebuah prototipe dan iterasi akhir menghasilkan perangkat lunak yang sudah lengkap. Gambaran yang merupakan tahapan dari model spiral sebagai berikut:

**

Sumber : **S, Rosa A dan M. Shalahuddin, 2018**

**Gambar 2.6 Model Spiral**

Ada beberapa tahap yang terdapat pada model spiral dimana pada tahapan tersebut terdiri dari :

1. Komunikasi dengan pelanggan (*customer communcation*)

Aktifitas membangun komunikasi yang efektif antar *developer* dan *customer*.

1. Perencanaan (*planning*)

Aktifitas untuk menjabarkan penjelasan sumber daya, waktu, dan informasi terkait terhadap proyek.

1. Analisis risiko (*risk analysis*)

Aktifitas untuk memprediksi risiko pada teknis atau manajemen.

1. Rekayasa (*engineering*)

Aktifitas untuk mendirikan satu atau beberapa representasi dari aplikasi perangkat lunak dapat berupa prototipe.

1. Kontruksi dan peluncuran (*construcion amd release*)

Aktifitas untuk mengonstruksi, menguji, melakukan instalasi, serta memberikan dukungan kepada *user*.

1. Evaluasi pelanggan (*customer evaluation*)

Aktifitas untuk menerima umpan balik dari evaluasi representasi perangkat lunak yang dihasilkan dari proses rekayasa perangkat lunak serta diterapkan pada proses instalasi.

Menurut (Simarmata, 2010:68) pada model spiral memiliki beberapa kerugian yang terdiri dari :

1. Penyesuaian terhadap perangkat lunak kontrakan

Perangkat lunak kontrakan berpusat pada kontrol, titik periksa (*checkpoint*), yang dapat dikirim dengan keadaan standar, hal itu keunggulan dari model *waterfall*. Model spiral sangat fleksibilitas dan kebebasan dan juga lebih pas dalam pengembangan perangkat lunak internal.

1. Berdasarkan terhadap keahlian manajemen risiko

Spesifikasi kendali risiko terdiri dari unsur risiko tinggi yang detail serta meninggalkan unsur untuk diteliti pada tahap selanjutnya. Hal yang perlu di pertimbangkan dimana spesifikasi yang dipegang risiko tergantung pada pengguna. Suatu rancangan yang telah dihasilkan dari seorang ahli akan diterapkan orang yang tidak berkompeten. Sang ahli harus menambahkan dokumentasi tambahan.

1. Kebutuhan dalam penelitian selanjutnya untuk langkah-langkah spiral

Model spiral menjelaskan bahwa memiliki proses yang fleksibel dan dinamis yang digunakan pengembang yang ahli untuk mendapatkan keuntungan maksimal. Untuk proyek berskala besar model spiral ada tahapan yang harus diteliti dan diimplementasikan secara khusus untuk menuju konsistensi, penjajakan,dan kontrol agar tercapai. Penelitian dan kontrol sangat penting, terkhusus analsis risiko dan manajemen risiko.

## 2.3 Flowchart

*Flowchart* merupakan gambar dan simbol yang berada pada suatu alur pemikiran dituangkan secara tertulis menyebabkan programmer dapat memberikan ide secara tertulis yang dapat dipahami oleh programmer lain, yang mana standar gambar dan simbol berdasarkan standar yang ada pada dunia komputer (Rahmat, Antonius , 2010 :13)

## 2.3.1 Penggunaan *Flowchart*

Ada beberapa simbol *flowchart* yang perlu diketahui berdasarkan kegunaan dari simbol tersebut yakni

**Tabel 2.1 Simbol dan *Flowchart***

|  |  |
| --- | --- |
| **Simbol** | **Keterangan** |
| Mulai / Start |  |
| Selesai / Finish |  |
| Aliran Data |  |
| Proses Kejadian |  |
| Sekuensial |  |
| Percabangan |  |
| Perulangan |  |
| Input / Masukan |  |
| Output / Keluaran |  |
| Pemberian Nilai / Inisialisasi |  |
| Memanggail Fungsi |  |
| Konektor pada halaman yang sama |  |
| Konektor untuk ditampilan halaman berbeda |  |
| Disk |  |
| Dokumen / Multi dokumen |  |

Simbol yang terdapat pada *flowchart* digunakan berdasarkan fungsi-fungsi yang tertera diatas yang memiliki penjelasan dalam pembuatan sebagai berikut :

1. *Flowchart* digambarkan lebih baik dari atas sampai kebawah dan dimulai dari bagian sebelah kiri terlebih dahulu.
2. Aktivitas yang dilakukan didalam *flowchart* harus dijelaskan secara detail.
3. Memberikan penjelasan awal *flowchart* dimulai dan sampai berakhir
4. Pada setiap langkah *flowchart* diberikan kata untuk mewakilkan suatu aktivitas yag dilakukan.
5. Segala aktivitas yang dilakukan harus secara berurutan.
6. Apabila kegiatan yang dilkukan terpotong, maka disambungkan ke tempat lain dengan menggunakan simbol penghubung.
7. Menggunakan simbol alir yang standar.

## Unified Modelling language (UML)

## 2.4.1 Definisi UML

UML (*Unified Modeling language*) adalah salah satu alat bantu yang sangat membantu dalam perkembangan sistem yang berorientasikan obyek. UML sangat mudah dipahami serta dilengkapi dengan mekanisme yang efektif untuk berbagi (*sharing*) dan mengkomunikasikan rancangan mereka dengan yang lain.

UML adalah kesatuan dari bahasa pemodelan yang dikembangkan oleh Booch, *Object modeling Technique* (OMT) dan *Object Oriented Software Enginering* (OOSE). Metode Booch dari Grady Booch sangat terkenal dengan nama metode design kedalam empat tahapan interatif, seperti identifikasi kelas-kelas dan obyek-obyek identifikasi semantik dari hubungan obyek dan kelas tersebut, perincian interface dan implementasi. didasakan pada analisis terstruktur dan pemodelan entity-relationship. Tahapan utama dalam metodologi ini adalah analisis, design sistem, design obyek dan implementasi.

* + 1. **Diagram UML**

Berikut ini diagram-diagram yang ada dalam UML :

1. *Use Case Diagram*

*Use case* merupakan deskripsi fungsi dari sebuah system dari perspektif pengguna yang mendeskripsikan tipikal interaksi antar user pada system di sistemnya sendiri melalui sebuah cerita bagaimana sistem digunakan. Langkah-langkah yang menjelaskan pengguna dan system dinamakan skenario. Setiap urutan diinisialisasi oleh orang, system yang lain, hardware atau urutan waktu.(Munawar, 2018a)

Berikut adalah simbol-simbol yang ada pada diagram *use case* :

**Tabel 2.2 Simbol – Simbol *Use Case* Diagram**

**Sumber :** (Shalahuddin and Rosa, 2018)

|  |  |
| --- | --- |
| **Simbol** | **Deskripsi** |
| *Use case* | Fungsionalitas sistem yang digunakan sebagai unit yang bertukar pesan dengan unit ataupun actor. Dinyatakan sebagai kata kerja diawal frase nama *use case* |
| Aktor / *actor*  Nama aktor | Aktor merupakan orang atau sistem yang melakukan kegiatan, meskipun simbol aktor merupakan gambar orang, tetapi aktor tidak selalu orang, biasanya dinyatakan sebagai kata benda diawal frasenama aktor. |
| Asosiasi / *association* | Asosiasi merupakan interaksi aktor dengan *use case* ataupun sebaliknya. |
| Ekstensi / *exstend*  *<<exstend>>* | Relasi *use case* tambahan kesebuah *use case*  dimana *use case* yang ditambahkan dapat berdiri sendiri. *Inheritance* pada pemogram berorientasi objek, biasanya  *use case* tambahan memiliki nama depan sama dengan *use case* yang ditambahkan, missalnya:  Arah panah mengarah pada *use case* yang ditambahkan. Biasanya *use case* yang menjadi *extend-*nya merupakan jenis yang sama dengan *use case* yang menjadi induknya. |
| *Generalisasi*/*generalization* | Hubungan *generalisasi* dan *spesialisasi* (umum-khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari yang lainnya, sebagai berikut:    Arah panah mengarah pada *use case* yang menjadi generalisasinya. |
| Menggunakan/*include/uses*  <<includee>>  <<uses>> | *Include* berarti *use case* yang ditambahkan akan selalu dipanggil saat *use case* tambahan dijalankan, misal pada kasus berikut :  *Include* berarti pengecekan terhadap use case tambahan apakah *use case* yang telah dtambah telah jalan sebelum *use case* dijalankan, misal pada kasus berikut;  Kedua interprestasi diatas dapat dianut salah satu atau keduanya tergantung pada pertimbangan dan interprestasi yang dibutuhkan. |

1. *Class Diagram*

*Class diagram* adalah diagram stastis. Ini mewakili pandangan statis dari suatu aplikasi. *Class diagram* tidak hanya digunakan untuk memvisualisasikan, menggambarkan, dan mendokumentasikan berbagai aspek sistem tetapi juga untuk membangun kode eksekusi (*executeble code*) dari aplikasi perangkat lunak. (Munawar 2018)

Simbol-simbol berikut ini merupakan simbol yang biasa digunakan pada pembuatan *clas diagram*, dapat kita lihat pada tabel berikut ini:

**Tabel 2.3 Simbol – Simbol *Class Diagram***

**Sumber : S, Rosa A. dan M. Shalahudin, 2013**

|  |  |
| --- | --- |
| **Simbol** | **Deskripsi** |
| Kelas   |  | | --- | | Nama\_kelas | | +*atribut* | | +*operasi()* | | Merupakan kelas yang ada pada struktur sistem |
| Antarmuka / *interface*    nama\_interface | Hubungan inteface antara sistem dengan aplikasi |
| *Asosiasi* / *association*  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Relasi antar kelas dengan makna umum, *asosiasi* biasanya juga disertai dengan *multiplicity* |
| *Asosiasi* berarah / *directed association* | Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, *asosiasi* biasanya juga disertai dengan *multiplicity* |
| *Generalisasi* | Relasi antar kelas dengan makna *generalisasi-spesialisasi* (umum-khusus) |
| Kebergantungan / *dependency* | Relasi antar kelas dengan makna kebergantungan antar kelas |
| Agregasi / *aggregation* | Relasi antar kelas dengan makna semua-bagian (*whole-part*) |

1. *Statechart Diagram*

*Statechart diagram* diagram menampilkan dua pandangan yang saling melengkapi *Statechart diagram* menunjukkan pesan-pesan yanng dilewatkan diantara obyek-obyek didalam sistem selama periode waktu yang pendek sedangkan state machine diagram akan mencari individu-individu obyek melalui keseluruhan hubungan dalam sistem, menspesifikasikan semua urutan yang mungkin dari pesan-pesan yang akan diterima obyek tersebut.(Munawar 2018)

Simbol-simbol berikut ini merupakan simbol yang biasa digunakan pada pembuatan *Statechart diagram*, dapat kita lihat pada tabel berikut ini :

**Tabel 2.4 Simbol-Simbol Statechart Diagram**

**Sumber :** (Shalahuddin and Rosa, 2018)

|  |  |
| --- | --- |
| Simbol | Deskripsi |
| Start / status awal (*initial state*) | Merupakan titik awal dari keadaan sistem yang dijalankan |
| End / status akhir (*final state*) | Merupakan titik akhir pada sistem |
| Event | Event merupakan kegiatan didalam sistem yang menyebabkan berubahnya status mesin |
| *State*  status | State atau status adalah keadaan sistem pada saat tertentu |

1. *Activity Diagram*

*Activity Diagram* merupakan bagian penting dari UML yang menggambarkan aspek dinamis dari sitem. Logika prosedural, proses bisnis dan aliran kerja suatu bisnis bisa dengan mudah dideskripsikan dalam acticity diagram .Peran *Activity diagram* hampir sama dengan peran *flowchart* , tetapi memiliki perbedaan *activity diagram* dapat mendukung perilaku paralel namun *flowchart* tidak bisa.(Munawar, 2018a)

Simbol-simbol berikut ini merupakan simbol yang biasa digunakan pada pembuatan *Activity diagram*, dapat kita lihat pada tabel berikut ini:

**Tabel 2.5 Simbol-Simbol Activity Diagram**

**Sumber : (Munawar 2018)**

|  |  |
| --- | --- |
| **Simbol** | **Keterangan** |
|  | Titik yang mengawali sistem |
|  | Titik yang mengakhiri sistem |
|  | Simbol Activity merupakan aktifitas pada sistem |
|  | Digunakan untuk menentukan pilihan |
|  | Fork : dipakai untuk menunjukan kegiatan yang dilakukan secara paralel atau untuk menggabungkan dua kegiatan paralel menjadi satu |
| Description: C:\Users\Toshiba\Music\1.png | Rake : menunjukan adanya dekomposisi |
| Description: C:\Users\Toshiba\Music\2.png | Tanda untuk menunjukan waktu |
| Description: C:\Users\Toshiba\Music\5.png | Tanda untuk menunjukan pengiriman |
| Description: C:\Users\Toshiba\Music\3.png | Tanda untuk menunjukan penerimaan |
| Description: C:\Users\Toshiba\Music\4.png | Tanda untuk Aliran akhir pada sistem (Flow final) |

1. *Sequence Diagram*

*Sequence diagram* digunakan untuk menggambarkan perilaku pada sebuah skenario. Diagram ini menunjukan sejumlah contoh obyek dan *message* (pesan) yang diletakan diantara obyek-obyek ini didalam *use case*.

Kompone utama squence diagram terdiri atas obyek yang dituliskan dengan kotak segiempat bernama. *Message* diwakili oleh garis dengan tanda panah dan waktu yang ditunjukan dengan *progress vertical.*(Munawar 2018)

Simbol-simbol berikut ini merupakan simbol yang biasa digunakan pada pembuatan *Squence diagram*, dapat kita lihat pada tabel berikut ini :

**Tabel 2.6 Simbol-Simbol Sequence Diagram**

**Sumber :** (Shalahuddin and Rosa, 2018)

|  |  |
| --- | --- |
| **Simbol** | **Deskripsi** |
| Aktor  atau | Aktor merupakan gambaran orang atau sistem lain yang dapat berinteraksi dengan sistem yang akan dibuat, meskipun aktor digambarkan sebagai orang namun aktor tidak selalu adalah orang. |
| Lifeline  **…………** | Menunjukan kehidupan suatu objek |
| Objek | Merupakan objek yang berinteraksi pesan |
| Waktu aktif | Menyatakan objek dalam keadaan aktif dan berinteraksi pesan |
| Pesan tipe create | Menyatakan suatu objek memuat objek yang lain, arah panah mengarah pada objek yang dibuat |
| Pesan tipe call | Menyatakan suatu objek memanggil operasi / metode yang ada pada objek lain |
| Pesan tipe send | Menyatakan suatu objek mengirimkan data / masukan / informasi ke objek lainnya |
| Pesan tipe return | Menyatakan objek telah melakukan operasi dan kembali ke objek tertentu |
| Pesan tipe destroy | Menyatakan bahwa objek akan masuk pada tahap mengakhiri suatu hidup objek yang lain |

1. *Collaboration Diagram*

*Communication diagram* (disebut *collaboration diagram* di UML) adalah sejenis diagram interaksi UML yang menunjukan interaksi antara obyek dan / atau bagian (direpresentasikan sebagai *lifeline*) menggunakan pesan berurutan dalam pengaturan dalam bentuk yang bebas. *Communication diagram* adalah perluasan dari obyek diagram yang menunjukan *massage-massage* obyek yang dikirimkan satu sama lain. Jadi *communication diagram* lebih menekankan pada *link* data diantara macam-macam *participant* pada interaksi tersebut. (Munawar, 2018a)

Simbol-simbol berikut ini merupakan simbol yang biasa digunakan pada pembuatan *Collaboration diagram*, dapat kita lihat pada tabel berikut ini :

**Tabel 2.7 Simbol-Simbol Collaboration Diagram**

**Sumber :** (Shalahuddin and Rosa, 2018)

|  |  |
| --- | --- |
| **Simbol** | **Deskripsi** |
| Objek  Description: Description: Description: Description: Description: Description: Description: Kelas peran | Objek yang melakukan interaksi pesan |
| Link | Relasi antara objek yang menghubungkan objek satu dengan lainnya atau dengan diri sendiri |
| Stimulus | Arah pesan yang terjadi, jika pada suati link ada dua arah pesan yang berbeda maka arah juga digambarkan dua arah pada dua sisi link |

1. *Deployment Diagram*

*Deployment diagram* menunjukan hubungan antara sebuah sistem secara fisik dan menampakan bagian bagian s*oftware* yang berjalan saling terhubung dengan bagian *hardware*.(Munawar, 2018a)

Simbol-simbol berikut ini merupakan simbol yang biasa digunakan pada pembuatan *Deployment diagram*, dapat kita lihat pada tabel berikut ini :

**Tabel 2.8 Simbol-Simbol *Deployment Diagram***

**Sumber :** (Shalahuddin and Rosa, 2018)

|  |  |
| --- | --- |
| **Simbol** | **Deskripsi** |
| Package | Package merupakan gabungan dari satu atau lebih node |
| Node  *Description: Description: Description: Description: Description: Description: Description: Node* | Node adalah sumber daya fisik yang menjalankan kode komponen |
| Dependency | Ketergantungan antara node, arah panah mengarah pada node yang dipakai |
| link | Relasi antara node |

## 2.5 Data Base

Menurut (Huda and Sutrisno, 2018) *database* adalah data yang dikumpulkan dan disimpan secara sistematis di dalam komputer yang dapat diolah atau diubah menggunakan *software* untuk mendapatkan informasi. Sedangkan *database* management system merupakan sistem atau software di mana sistem tersebut memungkinkan pengguna untuk mengontrol, mengakses, dan memelihara data secara praktis. *Database* berasal dari dua suku kata berupa data dan base, yang artinya berbasiskan pada data. tetapi secara *konseptual*, *database* dapat diterjemahkan berupa koleksi atau gabungan data yang berhubungan (*relation*), diatur berdasarkan aturan tertentu secara logis, maka akan menghasilkan sebuah informasi.

Secara prinsip dasar, *database* terdiri atas dua komponen penting, yakni data dan informasi. Basis data terdiri atas dua kata, yaitu basis dan data. Basis disebut juga sebagai markas atau pergudangan, tempat bersatu dan berkumpul. Sedangkan data suatu fakta mewakili objek baik manusia, peristiwa, konsep, keadaan, dan sebagainya, yang direkam dalam bentuk angka, huruf, simbol, teks, gambar, bunyi, atau kombinasinya menurut (Syam, Permana and Lusinia, 2018). Konsep dasar dari basis data adalah sekumpulan dari catatan (*note*), atau bagian dari pengetahuan. Basis data juga memiliki penjelasan terstruktur dari jenis fakta yang tersimpan di dalamnya yang disebut dengan skema. Skema pengidentifikasian obyek yang diwakili suatu basis data, dan hubungan di antara objek tersebut (Supriyatna, 2016) .

## 2.6 Sistem Manajemen Basis Data (DBMS)

Menurut (Ramakrishan dan Gehrke) dalam (Simarmata dan Paryudi, 2010) menyatakan sistem manajemen basis data merupakan perangkat lunak didesain dapat memudahkan dalam memelihara dan memanfaatkan sekumpulan dari data yang besar. Menurut (Kartika, Sovia and Sandawa, 2018) MySQL adalah *software* yang tergolong sebagai DBMS (*database management System*) yang bersifat *open source*.

## 2.6.1 Keuntungan dari DBMS

Ada beberapa keuntungan yang dimiliki yang berguna pada perusahaan di DBMS sebagai berikut :

1. Mengurangi pengulangan data

DBMS dapat mengurangi jumlah keseluruhan file dengan menghapus data yang terduplikasi dalam berbagai file.

1. Mencapai *independesi* data

Spesifikasi data disimpan dalam skema ditiap program yang ada pada aplikasi. Perubahan yang terjadi dapat dibuat pada struktur data tanpa mempengaruhi program yang melakukan akses data.

1. Mengintegrasikan data beberapa file.

File yang dibentuk dapat menyediakan kaitan logis, maka organisasi fisik tidak termasuk kedalam kendala. Organisasi logis, sudut pandang pengguna, dan program aplikasi tidak harus terdiri pada media penyimpanan fisik.

## 2.7 Entity Relationship Model (E-R *Model*)

## 2.7.1 Definisi Entity Relationship Model (E-R *Model*)

Pada *model entity-relationship* seluruh data *real* diterjemahkan dengan menggunakan beberapa alat konseptual menjadi diagram data yang disebut diagram E-R. Terdapat 2 komponen penting terbentuknya *model entity relationship* yaitu *entitas* (*entity*) dan relasi (*relation*) (Fathansyah, 2015:75).

Komponen tersebut dijelaskan lebih jauh dengan sejumlah atribut/properti sebagai berikut :

1. *Entitas* (*Entity*) dan Himpunan *Entitas* (*Entity Set*)

*Entitas* adalah individu yang mewakilkan hal yang *real* dan bisa dibedakan dari hal yang lain. Sekelompok *entitas* yang serupa dan berada dilingkungan yang sama disebut sebuah himpunan *entitas* (*entity set*). *Entitas* menujukkan pada individu yang terdapat pada objek sedangkan himpunan *entitas* menujukkan pada kelompok dari individu tersebut.

**Tabel 2.9 Tabel Himpunan Entitas Mahasiswa**

**sumber : Basis Data, Fathansyah, 2015**

|  |  |  |  |
| --- | --- | --- | --- |
| **Nim** | **Nama\_mhs** | **Alamat\_mhs** | **Tgl\_lahir** |
| 011234 | Ahmad | Jl. Melati 50 | 21-3-1980 |
| 011345 | Bobby | Jl. Mawar 103 | 13-5-1980 |
| 011456 | Charles | Jl. Mangga 145 | 17-8-1980 |

**Tabel 2.10 Tabel Kuliah**

**sumber : Basis Data, Fathansyah, 2015**

|  |  |  |  |
| --- | --- | --- | --- |
| **Kode\_mkuliah** | **Nama\_Mkuliah** | **sks** | **Semester** |
| 315KP2 | Pemrograman Visual | 2 | 5 |
| 317KP2 | Pemrograman Web | 2 | 5 |
| 319MP2 | Analisis Numerik | 2 | 5 |

1. Relasi (*Relationship*) dan Himpunan Relasi (*Relationship sets*)

Relasi merupakan adanya hubungan dibeberapa *entitas* dari himpunan *entitas* yang berbeda. Seluruh relasi terdapat pada *entitas*-*entitas* yang termasuk himpunan *entitas* membentuk himpunan relasi (*relationship set*).

**Tabel 2.11 Tabel Relasi Kedua Tabel Diatas**

**sumber : Basis Data, Fathansyah, 2015**

|  |  |  |
| --- | --- | --- |
| **Nim** | **Kode\_kuliah** | **Nilai** |
| 011234 | 315KP2 | A |
| 011234 | 319MP2 | B |
| 011345 | 315KP2 | C |
| 011345 | 317KP2 | A |
| 011345 | 319MP2 | C |
| 011456 | 317KP2 | D |
| 011456 | 319MP2 | B |

1. Atribut (*Attributesl Properties*)

Pada atribut memiliki karakteristik dari *entitas* tersebut. Pemilihan atribut yang relevan bagi *entitas* hal yang penting dalam pembentukan model. Penetapan atribut didasari pada tabel fakta, tetapi tidak hanya itu bahkan bisa dilihat dari proses normalisasi atau pemikiran tertentu, ada beberapa atribut yang tidak pada di ‘dunia nyata’ tapi perlu dimasukan yang selaras dalam membentuk model E-R terdapat pada kedudukan atribut di *entitas*. Harus dapat membedakan atribut yang berfungsi sebagai *primary key* dan mana tidak termasuk atribut deskriptif.

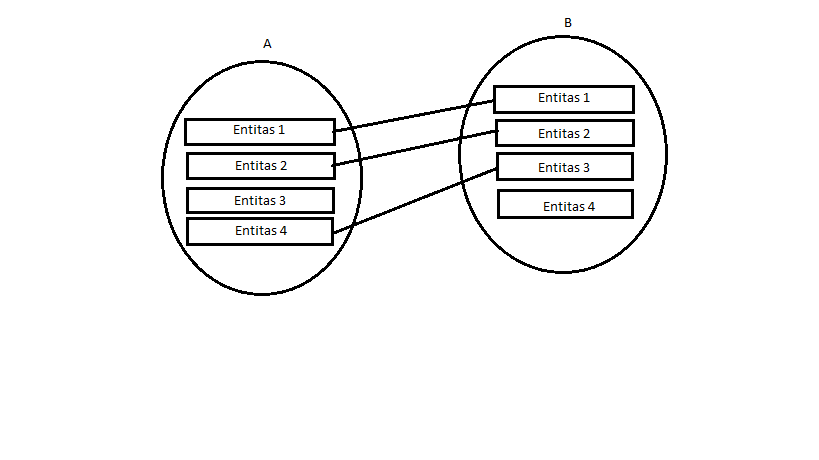
## 2.7.2 Kardinalitas/ Derajat Relasi

Kardinalitas relasi memperlihatkan nilai maksimum *entitas* yang berelasi dengan *entitas* pada himpunan lainnya. Kardinalitas relasi merujuk pada hubungan maksimum dari himpunan *entitas* satu dengan yang lain.

Kardinalitas relasi terjadi antara dua himpunan baik himpunan A dan himpunan B, dimana dapat dilihat seperti :

1. **Satu ke satu ( *one to one* )**

Pada kardinalitas relasi *one to one* tiap *entitas* himpunan A berhubungan paling banyak satu *entitas* himpunan *entitas* B, begitu juga sebaliknya.

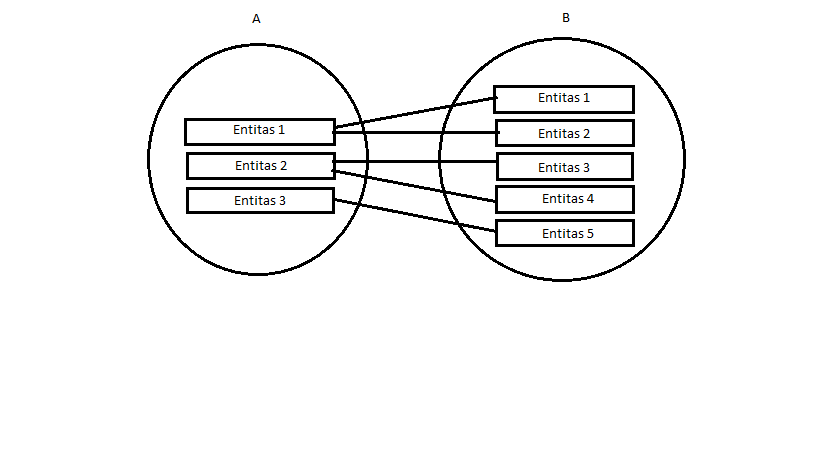


Sumber : **Fathansyah, 2015**

**Gambar 2.7 Kardinalitas relasi satu ke satu**

1. **Satu ke banyak ( *one to many* )**

Setiap *entitas* pada himpunan *entitas* A boleh memiliki hubungan dengan banyak *entitas* pada himpunan *entitas* B, tidak demikian pada *entitas* B yang hanya boleh memiliki hubungan dengan satu *entitas* himpunan A

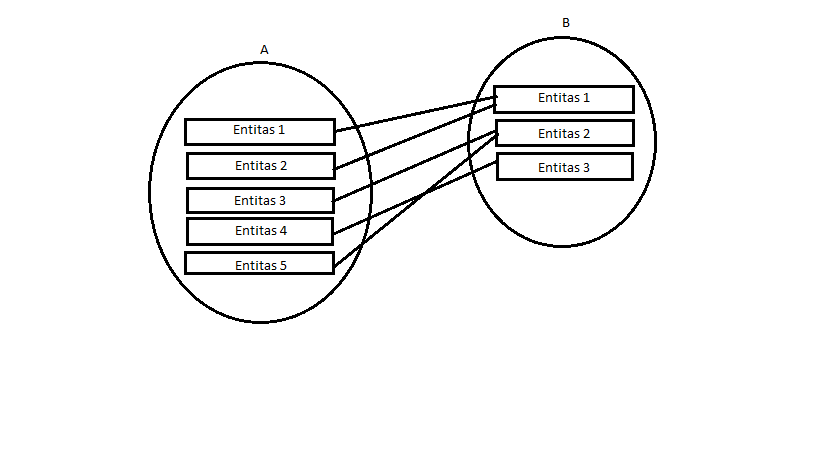


Sumber : **Fathansyah, 2015**

**Gambar 2.8 Kardinalitas relasi satu ke banyak**

1. **Banyak ke satu ( *many to one* )**

Setiap *entitas* pada himpunan *entitas* A terhubung paling banyak dengan satu *entitas* himpunan *entitas* B, tidak demikian pada himpunan *entitas* B yang mana setiap himpunan *entitas* B terhubung paling banyak satu *entitas* pada himpunan *entitas* A.

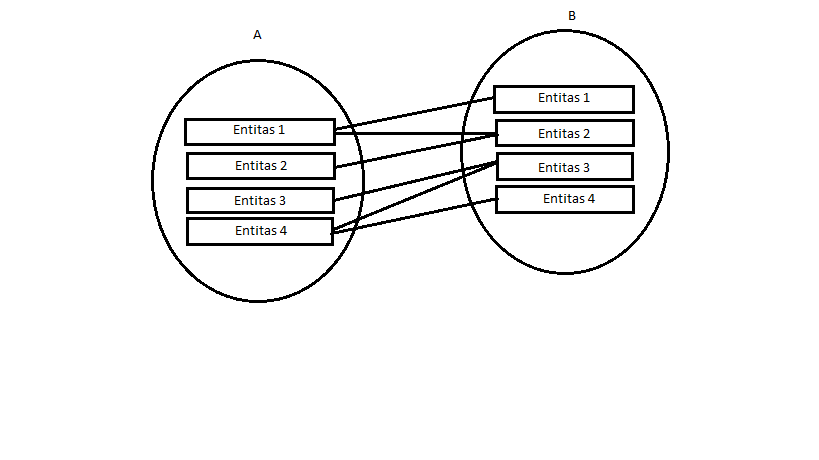


Sumber : **Fathansyah, 2015**

**Gambar 2.9 Kardinalitas relasi banyak ke satu**

1. **Banyak ke banyak ( *many to many* )**

Setiap *entitas* himpunan *entitas* A terhubung dengan banyak *entitas* himpunan *entitas* B, begitupun sebaliknya.



Sumber : **Fathansyah, 2015**

**Gambar 2.10 Kardinalitas relasi banyak ke banyak**

Kardinalitas relasi satu ke banyak dan banyak ke satu bisa dikategorikan sama disebabkan kardinalitas ditinjau dari dua sisi baik himpunan *entitas* A ke himpunan *entitas* B begiupun sebaliknya.

## 2.8 Entity Relationship Diagram (ERD)

Pada *entity relationship diagram* (ERD) menggambarkan secara sistematis komponen-komponen himpunan serta himpunan relasi yang terdapat pada model *entity relationship* ada atribut untuk direpresentasikan seluruh fakta.

Pada diagram E-R memiliki simbol-simbol notasi yang dapat digunakan didalamnya yaitu berupa :

1. Notasi simbol persegi panjang dinyatakan sebagai himpunan *entitas*.

E

Himpunan Entitas E

Sumber : **Fathansyah, 2015**

**Gambar 2.11 Himpunan *Entitas* E**

1. Notasi simbol lingkaran/elip dnyatakan atribut yang berfungsi sebagai *key*.

Atribut a sebagai Key

Sumber : **Fathansyah, 2015**

**Gambar 2.12 Atribut a**

1. Notasi simbol belah ketupat dinyatakan himpunan relasi.

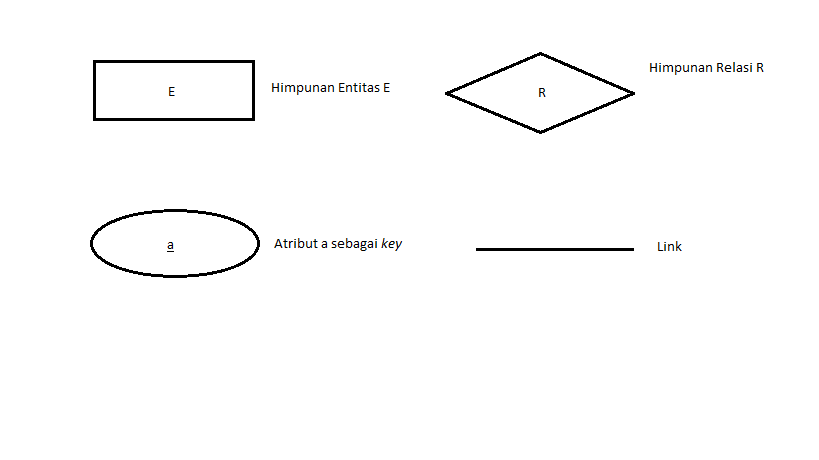
R

Himpunan Relasi R

Sumber : **Fathansyah, 2015**

**Gambar 2.13 Himpunan Relasi R**

1. Notasi simbol garis dinyatakan sebagai penghubung antar himpunan relasi terhadap himpunan *entitas* dan himpunan *entitas* terhadap atributnya.



Sumber : **Fathansyah, 2015**

**Gambar 2.14 Link**

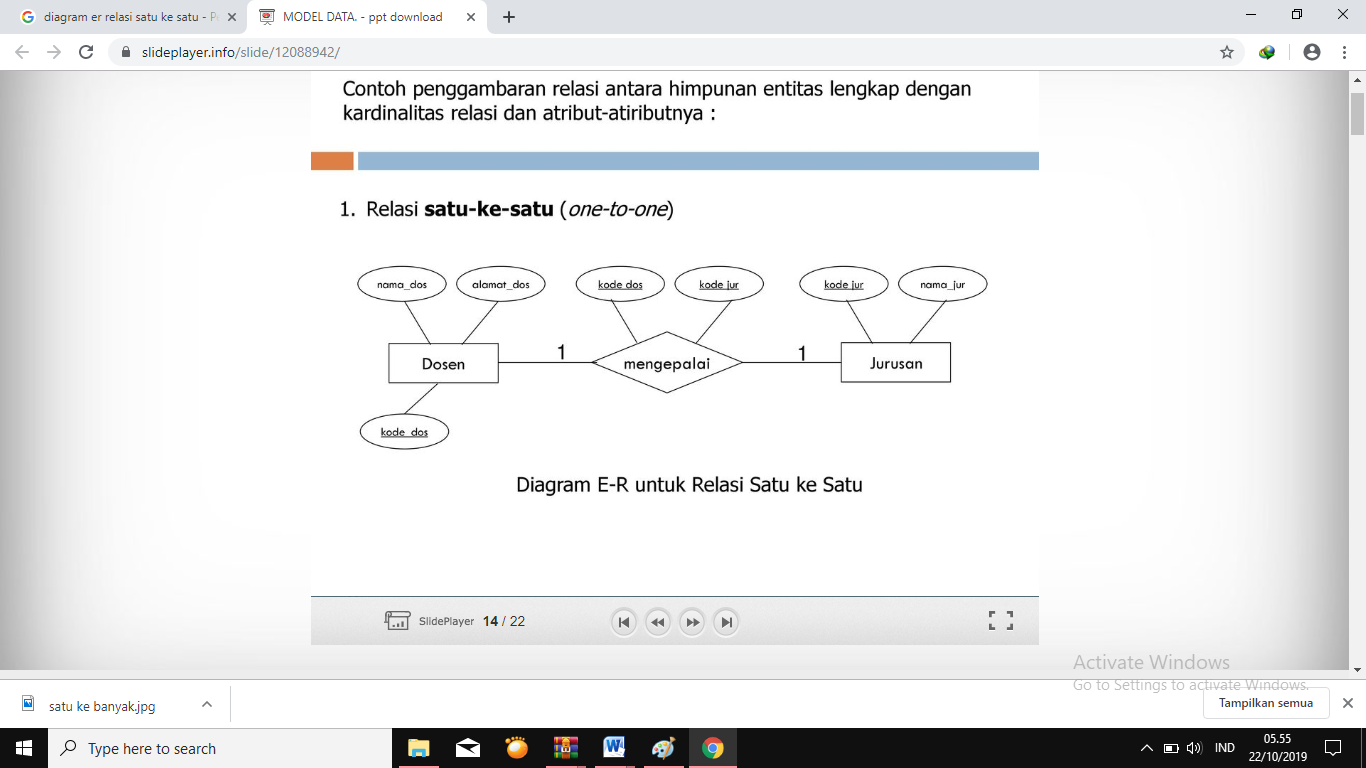
1. Kardinalitas relasi dapat diinisialkan dengan banyaknya garis cabang atau dengan menggunakan angka (1 dan 1 terhadap relasi satu ke satu, N dan N untuk relasi banyak ke banyak) .

## 2.8.1 Relasi Diagram Entity Relationship (ERD)

Pada *diagram entity relationship* memiliki gambaran relasi antar himpunan *entitas* terhadap kardinalitas relasi dan atributnya. Ada beberapa yang dapat dilihat contoh pada gambar dibawah sebagai berikut:

* 1. Relasi satu ke satu ( *one to one*)

Terdapat relasi antara himpunan *entitas* dosen dan jurusan, dimana himpunan relasinya diberi nama ‘Mengepalai’. Maka gambaran yang terdapat pada pernyataan diatas digambarkan sebagai berikut:



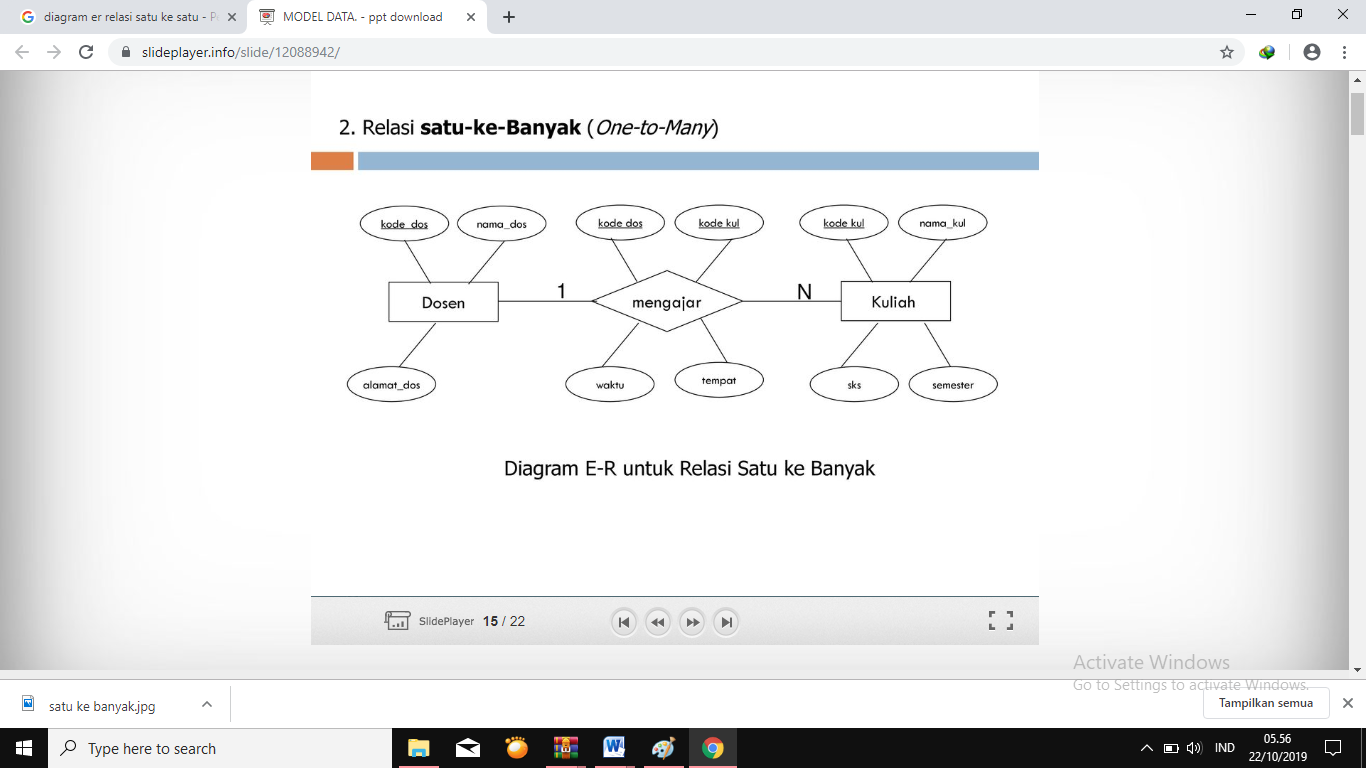
Sumber : **Fathansyah, 2015**

**Gambar 2.15 Relasi satu ke satu ( *one to one*)**

Pada diagram diatas bahwa himpunan *entitas* dosen dan jurusan terdapat 2 atribut. Sementara pada himpunan relasi mengepalai juga terdapat 2 atribut yang berfungsi sebagai *key* pada himpunan relasi. Kedua atribut berasal dari atribut key dari himpunan *entitas* yang terhubung, jadi keduanya dikelompokkan sebagai *key asing* (*foreign key*).

* 1. Relasi satu ke banyak (*one to many*)

Terdapat relasi himpunan *entitas* dosen terhadap himpunan *entitas* kuliah, yang mana himpunan relasinya bernama ‘Mengajar’. Maka dari pernyataan diatas dapat dilihat pada gambaran dibawah sebagai berikut:



Sumber : **Fathansyah, 2015**

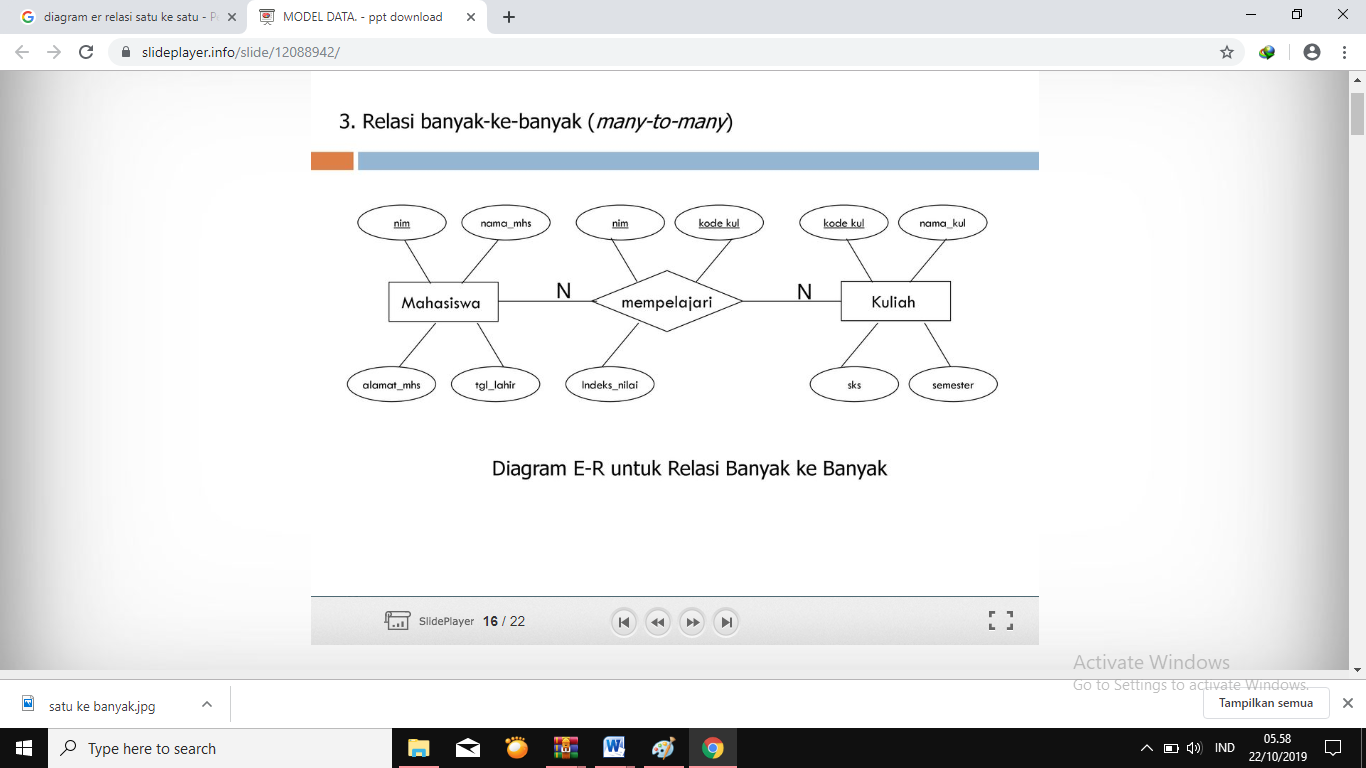
**Gambar 2.16 Relasi satu ke banyak ( *one to many*)**

*Foreign key* pada himpunan mengajar berupa kode\_dosen dan kode\_kuliah. Pada 2 atribut tambahannya bukan merupakan dari himpunan *entitas* dari 2 *entitas* tersebut.

* 1. Relasi banyak ke banyak ( *many to many*)

Terdapat relasi himpunan *entitas* mahasiswa dengan kuliah. Himpunan relasinya diberi nama ‘Mempelajari’. Himpunan relasi mempelajari terdapat 2 fungsi berupa menunjukkan mata kuliah apa yang akan diambil mahasiswa (atau mahasiswa mana yang akan mengambil mata kuliah tertentu) dan indeks nilai yang didapatkan mahasiswa untuk mata kuliah hal ini dapat teridentifikasi apabila data indeks nilai telah disimpan.

Dari penjelasan diatas, terdapat gambaran yang diberikan dibawah ini :



Sumber : **Fathansyah, 2015**

**Gambar 2.17 Relasi banyak ke banyak ( *many to many*)**

Dari diagram E-R terlihat bahwa adanya saling keterkaitan dimana relasi antar satu himpunan *entitas* yang lain dapat diketahui secara detail. Keberadaan relasi dapat kita duga secara intuitif dengan logika normal. Tetapi tiap relasi terlebih dahulu menentukan kardinalitas/derajat relasi serta atribut relasi tersbut. Dalam menentukan sangat sulit jika didasarkan dengan praduga atau terhadap evaluasi dari fakta yang ada. Oleh karena itu tanpa ada fakta yang memadai, keakuratan derajat relasi serta atribut relasi sukar kita jamin.

## 2.9 *Data warehouse*

## 2.9.1 Definisi *Data warehouse*

Penjelasan *data warehouse* memiliki banyak pengertian yang beraneka ragam menurut masing-masing para ahli, untuk lebih jelas dalam mendefinisikan *data warehouse* ada beberapa penjelasan dari para ahli. Menurut (Simon and Hammergren, 2009:29) *Data warehouse is a home for your high-value data, or data assets, that originates in other corporate applications, such as the one your company uses to fill customer orders for its products, or some data source external to your company, such as a database that contains sales information gathered from all your competitors.* Menurut (Simon and Hammergren, 2009:29) *Data warehouse* adalah tempat penyimpanan data bernilai tinggi atau aset data terpenting, data disimpan dalam sebuah aplikasi yang digunakan perusahaan untuk memenuhi pesanan pelanggan untuk produk-produknya, atau beberapa sumber data *eksternal* untuk perusahaan, seperti *database* yang berisi informasi penjualan yang dikumpulkan dari semua pesaing.

Definisi lain dari (Simon and Hammergren, 2009:33) *Data warehousing is the coordinated, architected, and periodic copying of data from various sources, both inside and outside the enterprise, into an environment optimized for analytical and informational processing. The keys to this definition for computer professionals are that the data is copied (duplicated) in a controlled manner, and data that is copied periodically (batch-oriented processing)*. Menurut (Simon and Hammergren, 2009:33) Penyimpanan data yang mencangkup keseluruhan data yang berskala besar yang merupakan sebagai penyalinan data yang terstruktur, dibangun, dan berasal dari berbagai sumber, yang terdapat di luar perusahaan, maupun berada di dalam lingkungan yang dioptimalkan untuk proses *analisys* dan informasi. Definisi ini untuk para ahli komputer bahwa data disalin (digandakan) secara terstruktur, dan data yang disalin secara berkala (pemrosesan berorientasi batch). M

Menurut (Rianto and Cucu Hadis, 2017) *Data warehouse* adalah suatu cara yang berguna untuk menampung dan menganalisis data berjumlah besar. Analisis tersebut dilihat dari beberapa dimensi atau sudut pandang yang berbeda.

Definisi *data warehouse* juga dikemukakan menurut (Inmon dan Richard) dalam (Mandala, 2016) memberikan sudut pandang bahwa, “*A Data warehouse is a subject oriented, integrated, time variant, and nonvolatile collection of data in support of management’s decision-making process*”. Menurut (Inmon dan Richard) dalam (Mandala, 2016) *Data warehouse* merupakan subjek yang berorientasi terintegrasi, memiliki varian waktu , serta memiliki pengumpulan data yang tidak mudah (*nonvolatile*) untuk memberikan dukungan dalam tahapan pengambilan keputusan manajemen.

Sementara menurut (Poe) dalam (Mandala, 2016) bahwa *Data warehouse* is read-only analytical database that is used as a foundation of decision support system. Yang menyebutkan bahwa (Poe) dalam (Mandala, 2016) memberikan pendapat *Data warehouse* adalah *database* analiti yang berkerja sebagai read-only yang digunakan pada dasar sistem pendukung suatu pengambilan keputusan.

Menurut (Fadilah, Winarno and Amborowati, 2018) *Data warehouse* merupakan data yang melalui tahapan dalam memilih jenis informasi yang dihasilkan, mengidentifikasi sumber data serta mentransfer informasi dari sumber data transaksional ke dalam *data warehouse* melalui tahapan *extract*, *transform* dan *loading* kemudian dari *data warehouse* ditransformasikan ke multidimensional data untuk keperluan analisis dan menyajikan informasi kepada *user* akhir dilingkungan manajemen tingkat atas.

## 2.9.2 Karakteristik Data warehouse

Pada *data warehouse* memiliki beberapa karakteristik yang terdapat didalamya.

Ada beberapa karakteristik *data warehouse* (Indrajani, 2011:279) yang terdiri dari :

1. *Subject-Oriented*

*Data warehouse* terdiri dari subjek atau disebut *entitas* bisnis utama, yakni pelanggan, pesanan atau produk. Pada orientasi subjek berbeda pada olahan transaksi yang lebih berorientasi proses.

Berikut merupakan tabel perbandingan *data warehouse* dengan OLTP :

**Tabel 2.12 Tabel Perbandingan *Data warehouse* dengan OLTP**

Sumber : **Indrajani,2011**

|  |  |
| --- | --- |
| **Sistem OLTP** | **Sistem *Data warehouse*** |
| Menyimpan data terkini | Data berbentuk historis |
| Menyimpan *detailed* data | Menyimpan *detailed*, *lightly*, *highly*, *summarized* data |
| Bersifat dinamis | Bersifat statis |
| Proses dilakukan berulang | Tidak terstruktur,*heuristic*,*processing* |
| *High level* dari *transaction throughput* | Medium ke *low level* dari *transaction* *throughput* |
| Pola berasal dari prediksi | Pola tidak bisa diprediksi |
| Lebih kearah transaksi | Orientasi subjek |
| Orientasi terhadap aplikasi | Pendukung keputusan strategis |
| Pendukung pada keputusan *all day* | Managerial user pada tingkat yang relatif rendah |
| *Operation user* jumlah besar | Lebih kearah *analysis* |

1. *Integrated*

Data yang operasional dari berbagai *basis data* dan sumber *data* *eksternal* yang diintegrasikan pada *data warehouse* untuk menghasilkan *basis data* tunggal untuk mendukung keputusan. Pegelompokkan data dibutuhkan *konvensi* penamaan yang konsisten, format data yang sama, dan skala perhitungan lintas *basis data* serta sumber *data* *eksternal* yang diperbandingkan.

1. Time-Variant

Pada *data warehouse* terdapat *time stamp* digunakan untuk merepresentasikan *data* *historys*. Dimensi waktu sangat kritis untuk mengidentifikasi trend,operasi-operasi dimasa depan, dan mengatur sasaran operasi. *Data warehouse* terdiri dari rangkaian *snapshot*, pada rangkaian merepresentasikan data operasional yang diambil di waktu tertentu.

1. *Nonvolatile*

Data baru yang ada akan ditambahkan, bukan digantikan sehingga data tersebut akan menjadi data *historys* yang tetap terjaga. Tindakan tersebut merupakan *update data warehouse*. Jila operasi *update* dan *delete* tidak ada, maka *data warehouse* bebas dari *anomali update* maupun *delete*.

## 2.9.3 Stuktur *Data warehouse*

Pada *data warehouse* memiliki struktur data yang dijelaskan dibawah ini menurut (Indrajani, 2011:280) yang mana struktur *data warehouse* terdiri dari :

1. *Current Detail Data*

Data yang terdiri dari level terendah pada struktur *data warehouse*. Pada data ini mendeskripsikan data detail yang aktif pada saat ini dan sedang berlangsung. Data ini membutuhkan media penyimpanan yang besar dan data yang sering diakses. Data ini merupakan data yang cepat diakses tetapi memiliki nilai harga yang relatif mahal dan kompleks dalam pemeliharaannya.

1. *Older Detail Data*

Data yang disebut sebagai data *back-up* (cadangan) yang sangat jarang diakses. Pada data ini disimpan pada media penyimpanan berbeda. Penyusunan direktori didasari urutan usia data dimana data tersusun rapi serta memberikan kemudahan dalam melakukan pengaksesan dilangkah selanjutnya.

1. *Lightly Summarized Data*

*Lightly Summarized Data*merupakan prosesdimanadata yang sering disebut sebagai ringkasan data dari *current detail data*. Tahap ini belum merupakan pengambilan sebuah keputusan disebabkan data belum memiliki sifat total *summary*, maksudnya data masih bersifat detail

1. *Highly Summarized Data*

*Highly Summarized Data* merupakan proses yang memiliki data bersifat *total summary*, ditahap ini data telah lancar untuk diakses terutama dalam melakukan perbandingan data yang beracuan pada urutan waktu dan analisis data telah menggunakan data multidimensi.

1. Metadata

Metadata adalah data tentang data yang merupakan penjelasan tentang struktur, isi, kunci, indeks dari data. Pada metadata memiliki pengelompokkan yang didalamnya terdiri dari :

1. *Technical metadata*

Tahap yang berisi informasi mengenai seluruh data dari *data warehouse* yang digunakan oleh administrator dan perancang *data warehouse* pada pengembangan *data warehouse* dan tugas manajemen.

1. *Businnes metadata*

Berupa informasi untuk *user* suatu *perspektif* agar dimengerti dari infromasi yang tersimpan pada *data warehouse.*

1. *Data warehouse operational infromation*

Data ini hampir seperti data *history* (*snapshot versions*), *ownership*, menjelajahi jejak audit, penggunaan data.

## 2.9.4 Aliran *Data warehouse*

Pada *data warehouse* memiliki aliran untuk melakukan proses sebagai berikut :

1. *Inflow*

Proses yang terdiri dari *ekstraction*, *cleansing*, dan *loading* data yang berasal dari sumber internal pada *data warehouse*.

1. *Uplow*

Proses yang menghubungkan penambahan nilai pada data di *data warehouse* yang diawali peringkasan, pemangketan, dan penyebaran data.

1. *Downflow*

Proses yang menghubungkan pengarsipan dan *backup* serta *recovery data warehouse*.

1. *Outflow*

Proses yang menghubungkan ketersediaan data bagi pengguna.

1. *Metaflow*

Proses yang terhubung dengan pengaturan metadata

## 2.9.5 Kegunaan *Data warehouse*

Pada *data warehouse* selain memiliki karakteristik terdapat kegunaan yang dimiliki oleh *data warehouse*. Menurut (Indrajani, 2011:283-284) terdapat beberapa kegunaan *data warehouse* yakni :

1. Pembuatan laporan

Pembuatan laporan salah satu kegunaan yang sangat umum, dengan menggunakan *query-query* sederhana yang dimiliki pada *data warehouse*, data dihasilkan per tahun, per bulan, per kuartal dan per hari. *Query* tersebut bertujuan untuk mendapatkan jawaban terhadap pertanyaan-pertanyaan khusus.

1. *On-Line Analytical Processing* (OLAP)

*Data warehouse* terdiri dari semua data serta infromasi mudah didapat, yang mana data tersebut dibutuhkan dalam tahap analisis data secara komplek. Serta melakukan analisis bisnis untuk mengetahui faktor kecenderungan pasar dan penyebabnya.

1. Data Mning

*Data warehouse* digunakan sebagai penentuan pola dan hubungan terhadap data satu dengan yang lain. Hal ini menyebabkan pembuatan *software* dibuat untuk pola statistik dalam data untuk mengetahui hal yang menyebabkan kecenderungan yang ada.

1. Proses Informasi Eksekutif

*Data warehouse* termasuk memiliki kegunaan dalam mencari ringkasan informasi penting yang memiliki maksud sebagai pengambilan keputusan tanpa mentelusuri seluruh data. Seluruh laporan telah diringkas dan diketahui detailnya hal tersebut memberikan kemudahan proses pengambilan keputusan yang menyebabkan data menjadi sangat informatif bagi user yang merupakan pihal eksekutif.

## Anatomi *Data warehouse*

Menurut (Indrajani, 2011:282-283) pada *data warehouse* memiliki 3 jenis dasar yang dijelaskan dibawah ini sebagai berikut :

1. ***Functional Data warehouse***

Fungsional adalah pendekatan kebutuhan dari tiap fungsi bisnis yakni departemen, divisi dan lainnya untuk menjelaskan jenis data yang ditampung terhadap sistem.Pendekatan ini diyakini untuk memberikan solusi yang mudah untuk dirancanng dengan biaya *investasi* yang relatif rendah dan dapat memberikan kemampuan sistem pengumpulan data yang terbatas terhadap kelompok pemakai. Penerapan pada jenis pengumpulan data bisa berakibat kehilangan konsistensi data diluar lingkungan fungsi bisnis. Apabila lingkup pendekatan ini diperbesar makan konsistensi data yang ada tidak dapat dijamin.

1. ***Centralized Data warehouse***

Pendekatan ini merupakan pendekatan yang baik karena keterbiasaan pemakai dengan lingkup *mainframe* terpusat. Data yang dimiliki berupa data seluruh sistem operasional dan disimpan di pusat penyimpanan data.

Keuntungan dari pendekatan ini dibanding dengan pendekatan fungsional yaitu data yang dimiliki benar-benar terpadu. Data pada pendekatan ini menharuskan data dikirim sesuai yang ditentukan agar tetap konsisten dengan pemasukan data yang lainnya. Tetapi apabila *user* mengambil data dari data terpusat dan tidak berkaitan secara langsung dengan pemasukan datanya sendiri.

Pada pendekatan ini memiliki pemeliharaan dengan harga yang sangat besar karena sistem pengumpulan data yang besar serta waktu yang diperlukan sangat lama untuk merancang sistem.

1. ***Distributed Data warehouse***

*Data warehouse* distribusi dikembangkan dengan konsep *gateway* *data warehouse* yang membuat *user* dapat terhubung dengan sumber data atau pemasok data maupun pusat data .

Pendekatan memakai teknologi *client* atau *server* untuk mengambil data dari sumber yang berbeda, hal ini menyebabkan tiap departemen atau divisi membangun sistem operasionalnya sendiri dan membangun data fungsional masing-masing dan menyatukan bagian tersebut dengan teknologi *client* atau *server*. Bila data menjadi konsisten maka pendekatan ini akan menjadi sangat efektif dan *user* bisa menambahkan data tersebut dengan informasi baru.Penerapan yang dilakukan sangat memerlukan biaya yang sangat besar hal ini disebabkan sistem operasinya dan pengumpulan datanya diolah dengan cara terpisah. Agar berfungsi bagi perusahaan, data terlebih dahulu disinkronisasikan demi memlihara keterpaduan data.

## 2.9.7 Manfaat *Data warehouse*

Menurut (Suni and Ridwan, 2018) manfaat pada pengimplementasi *data warehouse* merupakan berbagai *data heterogen* dari beraneka sumber bisa disimpan hanya dalam satu media penyimpanan menjadi *data homogen*. Data yang beraneka ragam dan tidak terstruktur diolah agar serupa menjadi bentuk yang sama dengan informasi yang sempurna. Menurut Sean Nolan dan Tom Huguelet, ada beberapa keuntungan yang dihasilkan dengan membentuk *data warehouse* yaitu keuntungan strategis untuk melebihi pesaing-pesaing dalam bisnis sebagai berikut:

1. Kemampuan akses data berskala besar.
2. Kemampuan terhadap data yang konsisten.
3. Kemampuan cepat dalam analisis.
4. Mengetahui terdapat hasil yang berulang-ulang dan sama.
5. Menemukan ruang pada *business knowledge* atau *business process*.
6. Mengurangi biaya administrasi
7. Memberikan wewenang setia anggota perusahaan dengan menyediakan informasi yang diperlukan agar kinerja bisa lebih efektif.

## ETL *(Extract, Transform, and Load)*

Pada tahapan *data warehouse* terdapat tahapan ETL (*extract, transform, and loading*). Menurut (Pratama, 2018) bahwa ETL (*Ekstrak, Transformasi,* and *Load*) merupakan proses yang sangat penting dalam mendirikan *data warehouse*. Proses ETL memiliki fungsi untuk mengumpulkan data dari berbagai sumber data yang berbeda-beda menjadikan seluruh kumpulan data yang terintegrasi. Pada penjelasan diatas merupakan urutan langkah-langkah dalam proses tahapan *data warehouse* yang melibatkan 3 proses, pada tahapan tersebut membuat proses terlihat lebih mudah dan mengerti disetiap proses data dalam jumlah banyak dari berasal dari berbagai sumber data. ETL sudah ada sejak tahun 2000 pada saat itu *database* memiliki peranan penting yang menerapkan teknologi informasi didalamnya.

## Proses Tahapan ETL *(Extract Transform and Load)*

Pada ETL ada langkah-langkah proses yang lakukan terlebih dahulu, dimana proses merupakan dari *extract, transform and load*. ETL merupakan proses yang dianggap komplek, disebabka pada setiap proses yang dilakukan terdapat beberapa proses didalamnya.

Berikut tahapan-tahapan proses ETL ( *extract, transform and loading*) menurut (Pratama, 2018: 228-234) sebagai berikut :

1. Tahapan*Extraction*

Pada *Extraction* pertama ini mengumpulkan semua data dari berbagai sumber data, sumber data yang didapat berasal dari berbagai *database* yang menggunaka OLTP (*Online Transaction Processing*), *website*, aplikasi, *flat file*, dan lain sebagaiannya. Tahap ini memiliki tujuan untuk mengidentifikasi bagian data yang tepat, data tersebut dimasukkan kedalam aliran (*work flow*) dari ETL untuk tahapan selanjutnya.

Tahap ini yang paling sederhana, karena data yang di *ekstrak* hanyalah data-data sebelum proses *ekstraksi*, yang memiliki perbedaan dengan data yang telah dieksekusi (*ekstrak*) sebelum data terdapat pada proses ETL. Pada tahap ini memiliki 3 proses yang dialami untuk memanipulasi *database* yaitu *insert* (penambahan data), *delete* (hapus data), dan *update* (pembaruan data).

Pada tahap ini sangat bergantung pada sistem secara keseluruhan, bahkan dalam *Idle Time*. Dimana pada kondisi *idle time* adalah kondisi yang relatif kosong (sepi maupun tidak aktif). Kondisi pertama yang dialami *overhead* terhadap sistem yang ada pada server. Hal ini terjadi akibat kondisi  *idle time*, seluruh tahap administratif dan proses ekstraksi data dilakuka oleh *server*. Kondisi kedua yang terdapat pada *overhead* membuat munculnya *intervensi* (campur tangan) perangkat lunak komputer (*software*) secara keseluruhan. Proses selanjutnya yaitu *change data capture*. Pada tahap ini proses untuk mengambil beberapa kebijakan (*policies*) yang tidak sama untuk kepentingan ektraksi data. Untuk hasil yang maksimal dilakukan rekaman data (*data snapshot*) yang juga diekstraksi .hasil tersebut dibandingkan dengan data snapshot yang berasal dari berbagai data sumber atau dari DSA ( *Data staging area*).

Keluaran atau disebut juga dengan output merupakan sekumpulan data dari berbagai jenis sumber data yang dimasukkan ke format data. Pada data yang belum memiliki format data yang belum sama, selain dari berbagai sumber data, juga belum dilakukan dari proses pemformatan ulang yang telah disepakati bersama pada sistem.

1. Tahapan *Transformation*

Tahapan *Transformation* ini memiki proses untuk melakukan penyeragaman format. *tranformation* yang dilakukan dari ETL Tool yang terdapat dari ETL itu sendiri.

Ada 2 langkah yang dilakukan untuk penyeragaman format data. Pertama *Look Up Bulk Data* (LUBD) yang memuat data dari berbagai macam *record* dari *database* ataupun data dari berbagai sumber data. LUBD langkah untuk mencari *bulk data* yang berasal dari tahap *extraction*  yang diteruskan kedalam tahap transform. Langkah kedua yaitu *transform* yang mentransformasikan format data yang berbeda-beda dari seluruh data yang didapatkan dari sumber data kedalam format data tunggal yang telah disepekati.

1. Tahapan*Loading*

Tahapan data telah diseragamkan secara keseluruhannya kedalam format data yang telah disepakati. *User* mengakses *output* dari tahapan *loading* dalam bentuk data informasi. Penyajian yang dilakukan dalam bentuk laporan (*reporting*) atau sekelompok data demi kebutuhan analisis dan pengambilan keputusan. Pada tahapan ini memiliki 3 proses yang ada. Yang pertama *Load Up Data* (LUD) untuk menyalurkan data yang telah diintegrasikan dan diseragamkan kedalam tatap muka aplikasi ataupun tempat dimana melakukan pengaksesan. Tahapan yang kedua yaitu *Load Insert Data* (LID) dimana data dimasukkan menuju kedalam databse pada *data warehouse*. DBMS dan *database* telah mendukung format data yang telah disepakati bersama. Tahapan terakhir yaitu *Load Bulk Data* (LBD) untuk melanjutkan

*Bulk Data* yang terdapat data-data dan informasi yang merupakan hasil penggabungan (*integrasi*) yang telah diproses pada *transformation*.

## Metode Perancangan Basis Data Untuk *Data warehouse*

Menurut (Connolly and Begg, 2005) terdapat 9 langkah metode Kimball atau *nine-step* Kimball diperkenalkan oleh ahli *data warehouse* dan *business intelligence* yang bernama Ralph Kimball (1944) yang digunakan untuk langkah-langkah membangun dan mengembangkan suatu *data warehouse*.

* + - 1. Menentukan proses bisnis (*Choosing the process*)

Proses ini cenderung menjadi salah satu yang terkait dengan penjualan. Sumber data ini dapat memungkinkan diakses dan berkualitas tinggi. Data yang akan dibangun haruslah yang memiliki data berupa anggaran, dan data yang berhubungan dengan bisnis yang paling penting secara komersial. Pertama-tama kami mengidentifikasi bahwa proses bisnis.

* + - 1. Menentukan *granularity* (*Choosing the grain*)

Menentukan *granularity* berarti memutuskan dengan tepat tabel fakta. Untuk menentukan apa yang diwakili secara pasti maupun direpresentasikan oleh suatu tabel fakta. Ketika menentukan *granulaty* untuk tabel fakta berarti dapat kita mengidentifikasi dimensi tabel fakta. *Entitas* yang akan digunakan untuk referensi data akan menjadi tabel dimensi sebuah skema. Hasil akhir dari tabel fakta juga menentukan dari masing-masing dimensi. Contohnya, jika sumber dari sebuah tabel fakta properti *sale* adalah properti *sale* pribadi maka sumber dimensi pelanggan adalah mengenai rincian pelanggan yang membeli properti utama.

* + - 1. Identifikasi dan penyesuaikan dimensi (*Identifying and conforming the dimensions*)

Untuk identifikasi dimensi dalam mengatur konteks tentang fakta di tabel fakta. Dimensi yang dibangun dengan baik membuat akan dapat dimengerti dan mudah digunakan. Identifikasi dimensi yang dibuat baik dimensi yang didapat buruk atau tidak lengkap akan mengurangi kegunaan *data mart* atau *data warehouse* untuk suatu perusahaan. Dimensi merupakan hal yang penting untuk menggambarkan mengenai fakta-fakta yang ada pada tabel fakta. Jika terdiri dimensi yang muncul pada dua *data mart*, maka kedua *data mart* perlu memiliki dimensi yang sama. Atau setidaknya salah satunya harus berbentuk subset matematis. Dan jika pada dimensi dipergunakan dua *data mart* atau lebih dan dimensi tidak dapat sinkron maka seluruh *data warehouse* akan mengalami kegagalan karea dua *data mart* tidak dapat dipergunakan secara bersamaan.

* + - 1. Menentukan fakta (*Choosing the fact*)

Untuk menentukan fakta mana yang dapat digunakan dalam *data mart* atau *data warehouse*. Semua fakta harus dibuat secara detail. Fakta harus numerik dan aditif. Fakta tambahan dapat ditambahkan ke tabel fakta kapan saja asalkan konsisten pada tabel fakta sebelumnya. Sumber dari tabel fakta menentukan fakta yang dapat digunakan pada *data mart*, Seluruh fakta harus diekspresikan pada tingkat yang sudah ditetapkan oleh sumber. Melakukan penyimpan prekalkulasi pada tabel fakta apabila fakta kehilangan statement dan melengkapi data dimensi.

* + - 1. Menyimpan hasil perhitungan sementara pada tabel fakta (*Storing pre-calculations in the fact table*).

Tabel fakta dipilih, masing-masing harus diperiksa ulang untuk menentukan apakah ada adalah peluang untuk menggunakan pra-perhitungan. Fakta ini berguna karena bernilai jumlah *adiktif*, yang dapat kita peroleh informasi berharga seperti rata-rata alokasi berdasarkan jumlah *agregasi* catatan tabel fakta.

* + - 1. Melengkapi tabel-tabel dimensi (*Rounding-out the dimension tables*)

Dalam langkah ini dapat menambahkan tabel dimensi dan menambahkan deskripsi teks ke dimensi yang mungkin ada. Deskripsi teks harus intuitif dan dapat dimengerti. Kegunaan *data mart* atau *data warehouse* ditentukan oleh ruang lingkup dan sifatnya dari atribut dari tabel dimensi.

* + - 1. Menentukan durasi dimensi (*Choosing the duration of the dimension*)

Untuk menentukan durasi mengukur seberapa jauh ke belakang waktu tabel fakta berjalan. Untuk melihat periode waktu yang sama satu atau dua tahun sebelumnya. Untuk menyimpan data memperpanjang kembali lima tahun atau lebih. Tabel fakta yang sangat besar memunculkan setidaknya dua yang sangat signifikan masalah desain *data warehouse*.

1. Menelusuri perubahan dimensi secara perlahan (*Tracking slowly changing dimension*)

*Data warehouse* harus menetapkan *primary key* untuk dimensi penting ini secara berurutan untuk selama periode waktu tertentu.

Ada tiga jenis dasar dimensi yang dapat berubah sewaktu-waktu:

* + - 1. Perubahan atribut dimensi apabila mengalami kesamaan .
      2. Atribut dimensi yang berubah menyebabkan dimensi baru akan mengalami penambahan pada dimensi baru yang akan dibuat.
      3. Atribut dimensi berubah menyebabkan atribut alternatif dibuat sehingga nilai-nilai lama dan baru dari atribut tersebut secara bersamaan dapat diakses dalam dimensi yang sama.

1. Memutuskan prioritas *query* dan bentuknya (*Deciding the query priorities and the query modes*)

Pada langkah ini kami mempertimbangkan masalah desain fisik. Model dimensi, yang terdiri lebih dari satu tabel fakta dari berbagi tabel dimensi yang sesuai, disebut sebagai konstelasi fakta.

## 2.11 Permodelan (*Schema*)

## 2.11.1 Star schema

Skema bintang adalah struktur logikal yang mempunyai tabel fakta terdiri atas data faktual ditengahnya, serta dikelilingi atas tabel-tabel dimensi yang berisi referensi data. *Dimension* *Table* merupakan tabel berisi kategori dengan ringkasan data detail yang bisa dilaporkan. Laporan keuntungan pada tabel fakta dapat dilaporkan sebagai dimensi waktu (yang berupa perbulan, perkwartal dan pertahun). *Fact* *Table* adalah tabel yang pada umumnya mempunyai angka dan data *history* dimana *key* (kunci) yang didapatkan sangat unik, karena *key* tersebut terdiri dari *foreign key* (kunci asing) adalah *primary key* (kunci utama) dari beberapa dimension tabel yang saling terhubung.

Menurut (Indrajani, 2011:285-286) berpendapat semua tabel memiliki *primary key* yang bisa mempunyai satu kolom atau lebih. *Primary key* tersebut membuat menjadi berbeda dan berkarakteristik. Pada tabel fakta *primary key* terdiri atas satu atau beberapa *foreign key*. Pada kolom tabel nilai dijelaskan oleh *primary key* pada tabel yang lain (Rahutomo *et al.*, 2019).

Terdapat gambar dari skema bintang berdasarkan penjelasan diatas sebagai berikut :

Dimensi 5

Fakta

Dimensi 1

Dimensi 2

Dimensi 3

Dimensi 4

Sumber : (Henderi and Winarko, 2015)

**Gambar 2.18 *Star schema***

## 2.11.1.1 Kelebihan dari *Star schema*

Pada *star schema* memiliki beberapa kelebihan yang terdapat didalamnya yakni sebagai berikut :

* + - 1. Memberikan kemudahan bagi *developer* untuk memahami dalam pemodelan serta desain sebelum diimplementasikan kedalam sistem.
      2. Proses *querry* yang dilakukan lebih cepat dan tidak menghabiskan waktu serta sumber daya yang banyak.
      3. Memudahkan dalam memprediksi *querry*, yang menyebabkan penambahan atribut pada tabel dimensi.
      4. Mampu menampilkan *logic* informasi dan memberikan informasi lebih baik, cepat, dan simpel.
      5. Pada OLAP, *star schema* cocok digunakan dalam *data multidimensional*.
      6. Memudahkan dalam *development*.
      7. Dapat menyesuaikan berdasarkan kebutuhan dan perubahan yang terjadi pada organisasi.
      8. Dapat menjabarkan situasi dari suatu organisasi dalam bisnis.
      9. Memudahkan dalam proses *navigasi* agar mudah dipahami oleh pengguna, mudah dipahami dan dapat digunakan dan hal ini meyebabkan *star schema* lebih optimal.

## Kekurangan dari *Star schema*

Selain memiliki kelebihan *star schema* juga memiliki kekurangan yang ada sebagai berikut :

1. Terjadinya pemborosan terhadap ruang (*space*) pada desain dan pengimplementasiannya.
2. Ukuran penyimpanan yang berskala besar.
3. Akan mengalami kesulitan dalam melakukan perawatan dan perbaikan.
4. Lebih mudah rentan terhadap data yang dapat diduplikasi.

## 2.11.2 Snowflakes Schema

Menurut (Connolly dan Begg) dalam (Pratama, 2018: 215) *snowflakes* *schema* adalah permodelan pada data multidimensi yang merupakan perluasan dari *star schema*, yang memiliki ciri tidak semua dimensi terdapat pada tabel fakta tetapi dpat terhubung pada dimensi utama.

Dimensi 1

Tabel Fakta

Dimensi 2

Dimensi 3

Dimensi 6

Dimensi 5

Dimensi 4

Sumber (Seta, Wati and Isnainiyah, 2017)

**Gambar 2.19 *Snowflakes* *Schema***

## Kelebihan Snowflakes Schema

Adapun penjelasan diatas, *snowflakes* *schema* juga memiliki kelebihan yang terdapat pada skema tersbut yakni sebagai berikut :

1. Kapasitas yang dimiliki lebih sedikit yang memberikan penghematan dalam ruang penyimpanan.
2. Memudahkan dilakukannya perbaikan dan perawatan serta pengembangan.
3. Skema ini sangat cocok untuk proses *querry* ETL.

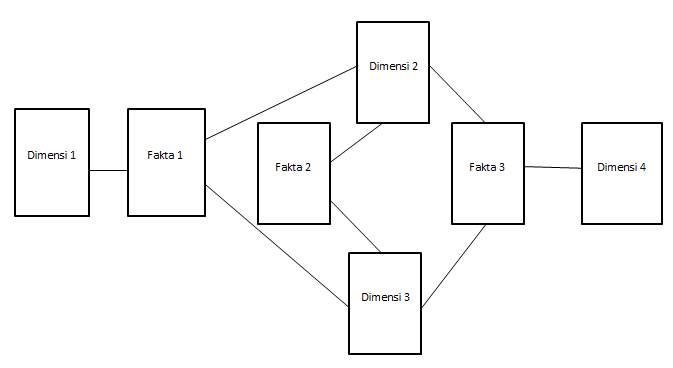
## Kekurangan Snowflakes Schema

*Snowflakes* *schema* memiliki kekurangan yang terdapat pada skema tersebut selain memiliki kelebihan yang telah dijelaskan. Kekurangannya sebagai berikut :

1. Performansi masih termasuk lambat pada skema tersebut.
2. Masalah komplesitas pada skema yang menyebabkan user sulit dalam mengenali isi dari skema tersebut, karena skema ini bersifat *intuitif* yaitu hanya memperlihatkan tujuan secara lebih jelas dan tidak tampi secara eksplisit.
3. Terjadinya masalah komplesitas yang membuat pengguna sulit memperoleh informasi dan kejelasan dari model tersebut.

## 2.11.3 Starflakes Schema

*Starflakes* m*schema* merupakan gabuangan dari *star schema* dan *snowflakes* *schema* yang menyebabkan tidak semua dimensi termasuk kedalam tabel fakta dan hanya cukup pada tabel dimensi utama, tabel dimensi telah dinormalisasikan dan ada pula yang terdenormalisasi. *Starflakes* *schema* berasal karena perbaruan dua *schema* sebelumnya.



Sumber : (Aljawarneh, 2016)

**Gambar 2.20 *Starflakes* *Schema***

## 2.11.3.1 Kelebihan Starflakes Schema

*Starflakes* *schema* memiliki kekurangan yang dimilikinya yaitu sebagai berikut:

1. Mudah beradaptasi terhadap kebutuhan perusahaan atau organisasi.
2. Memudahkan *user* dalam melakukan akses data dan mengefisiensikan waktu serta sumber daya komputasi untuk pengkasesan data.

## Kekurangan Starflakes Schema

*Starflakes* *schema* selain mempunyai kelebihan juga memiliki kekurangan yang terdapat didalamnya yakni tidak fokus didalam proses data yang terjadi akibat menggabungkan normalisasi dan denormalisasi pada data multidimensi, sehingga masalah redudancy pada data tersebut diselesaikan outtriger. Tetapi pada kekurangan ini masih bersifat relatif.

## OLAP dan OLTP

## OLAP (Online analytical processing)

OLAP (*Online analytical processing*) adalah salah satu cara untuk pengolahan data pada *data warehouse*. Proses OLAP dapat digunakan sebagai media untuk memberikan informasi secara detail dan terperinci tentang informasi strategis yang dibutuhkan oleh *user*. Proses OLAP yang digunakan antara lain *roll up* & *drill down* (Khotimah and Sriyanto, 2017). OLAP yaitu sintesis, konsolidasi, dan analisis dinamis dari sejumlah besar data multidimensi.

Menurut (Huda and Sutrisno, 2018) OLAP secara keseluruhan berupa aturan pada sebuah kerangka untuk mendukung sebuah keputusan. Menurut (Connolly dan Begg) dalam (Supriyatna, 2016) *Online analytical processing* yakni gabungan dari dinamis analisis dan juga dari data multidimensional berskala besar. OLAP terdiri dari gabungan aturan yang menyediakan sebuah kerangka dimensional untuk mendukung pengambilan keputusan. OLAP juga sebuah pendekatan yang cepat dalam menyediakan jawaban terhadap *querry* analitik yang multidimensi.

OLAP adalah bagian jenis yang lebih global dari pemikiran bisnis, yang mencangkup hubungan antara pelaporan dan penggalian data. Pada sistem OLAP konsep dari sebuah kubus OLAP (disebut juga sebagai kubus multidimesi atau *hiperkubus*) yang terdiri dari numeric fact yang disebut ukuran dan dikategorikan sebagai dimensi. Kubus metadata secara khusus berasal dari skema bintang atau skema kristal salju (*snowflake*) dari tabel *database* yang berhubungan. Ukuran diturunkan dari rekord dalam fact table dan dimensi-dimensi yang diturunkan dari tabel-tabel dimensi (Supriyatna, 2016). OLAP merupakan perangkat yang menjelaskan teknologi dengan menggunakan visualisasi multidimensi sejumlah data bertujuan untuk lebih cepat dalam analisis. Model data OLAP, Menjelaskan informasi secara konseptual seperti kubus (cube), yang terdiri atas kategori deskriptif (dimensions) dan nilai kuantitif (measures) (Rianto and Cucu Hadis, 2017).

## 2.12.1.1 Jenis-jenis pada OLAP (Online analytical processing)

Olap memiliki 3 kriteria yang digunakan pada *data warehouse*. Kriteria tersebut terdiri sebagai berikut:

1. Rolap (Relational OLAP)

Rolap merupakan data relasional yang berasal dari sumber-sumber data dengan menggunaka RDBMS dan data tersimpan dalam bentuk rasional. Rolap proses untuk melakukan analisis serta dapat memanipulasi data.

1. Molap (Multidimensional OLAP)

Molap merupakan tempat penyimpanan data yang terstruktur multidimensi dengan menggunakan *array* sebagai pemanfaatannya. Molap diimplementasikan kedalam bentuk *data cube* yaitu data yang memanfaatkan bentuk virtual bentuk ruang kubus untuk memvisualisasikan data-data multidimensi.

1. Holap (Hybrid OLAP)

Holap memiliki kelebihan serta manfaat yang lebih dibandingkan jenis-jenis sebelumnya, pada holap dapat menyimpan data pada dua jenis DBMS yang berupa RDBMS (*Relational Database Management System)* sebagai data yang bersifat rational dan MDDBMS (*Multidimensional Database Management System*) sebagai data yang bersifat multidimensi. Pada dasarnya holap sering menggunakan RDBMS sebagai penyimpanan data dan menggunakan MDDBMS sebagai agregasi data untuk pemanfaatannya.

## 2.12.2 OLTP (Online analytical processing)

OLTP sebuah sistem yang berkemampuan untuk melakukan pencatatan transaksi berbasis komputer dan internet, mempunyai *querry* yang sederhana, *fast* terhadap respon yang didapatkan, dan terdesentralisasi. OLTP memiliki kegunaan untuk melakukan pencatatan, melakukan proses, melakukan pengolahan, dan menghasilkan data dan informasi mengenai transaksi sutau kejadian (Pratama, 2018).

## 2.12.3 Perbedaan OLAP dan OLTP

Menurut (Pratama, 2018) OLAP dan OLTP memiliki beberapa berbedaan yang dapat dilihat sebagai berikut :

1. Sumber Data ( *Data Source*)

Data yang terdapat pada OLTP bersifat *Operational Data*. *Operational data* dapat diartikan sebagai pengumpulan data yang diambil dari berbagai sumber data utama (*original source*), yang melibatkan data opersional dan data transaksi yang disatukan dalam sebuah *database* sebagai media pengumpulnya, data yang didapat bersumber keaslian atau berasal dari data utama yang mana data berformat dari beraneka ragam.

1. Tujuan

OLTP memiliki tujuan melakukan pemrosesan kontrol dan lajunya proses bisnis serta tugas fundamental di suatu organisasi. OLAP memiliki tujuan yang lebih luas, oleh sebab itu olap lebih sering dan digunakan dalam proses transaksi yang menyebabkan OLAP diimplementasikan pada sistem *database* yang berskala besar.

1. Kecepatan Pemrosesan Data

OLTP memiliki kecepatan yang tinggi dalam proses data, sedangkan OLAP memiliki kecepatan yang dapat ditinngkatkan dengan melakukan pembuatan index. Dengan ini dapat melakukan pembantuan dalam pembuatan querry, akan tetapi pengaruh kecepatan OLAP terdapat dari faktor yang lain seperti jumlah data, volume data, *kompleksitas querry*, dan *batch data*.

1. Jenis Data

OLTP lebih kearah data yang bersifat dua dimensi, yang menyebabkan data memiliki kolom (*field*) dan baris (*record*) dengan sebuah *primay key* pada tiap tabel. OLTP lebih mefokuskan pada proses bisnis yang sedang berjalan yang menyebabkan data *historys* terabaikan. OLAP memiliki data yang cenderung bersifat multidimensi, pada dasarnya multidimensi melakukan proses bisnis lebih dari satu dan melakukan banyak sudut pandang dari berbagai dimensi. OLAP memperhatikan data histori yang ada, sehingga pada data sebelumnya tidak ada penumpukkan / penggantian data.

1. Desain *Database*

OLTP memiliki desain *database* tabel yang dinormalisasikan, hal ini diharapkan untuk mengurangi duplikasi data (*redudancy*). Tabel OLTP saling berelasi satu sama lain. OLAP memiliki desain *database* yang disesuaikan dengan data multidimensi yang dimilikinya sebagai jenis data pada OLAP. OLAP tidak memiliki konsep normalisasi, hal ini menyebabkan OLAP menggunakan desain *database* seperti *Star schema*, *Snowflakes Scehma*, dan *Starflakes Schema*.

1. *Back Up* dan *Recovery*

OLTP melakukan *back up* dan *recovery* secara berkala dan teratur. OLTP dapat melakukan hal tersebut dikarenakan berorientasi kepada transaksi dan operasional yang berjalan terus menerus. Hasil akhirnya akan disimpan kedalam *database* OLTP serta dapat meningkatkan volume secara signifikan. OLAP tidak selalu melakukan *back up* dan *recovery*, hal ini dilakukan karena data pada OLAP berasal dari OLTP. Jadi apabila data dibutuhkan, cukup mengambil salinan dari OLTP.

1. Kapasitas (*Space*)

OLTP memerlukan kapasitas ruang yang lebih kecil dibandingkan dengan OLAP. Karena data pada OLTP lebih kepada transaksi dengan mengabaikan data *historys*. OLAP membutuhkan ruang penyimpanan yang begitu besar dikarenakan untuk melakukan agregasi data, pengindeksan data yang kompleks, analisis data, dan penyimpanan data *historys* yang utama pada OLAP.

1. *Query*

OLTP memiliki *query* yang sederhana, standar, dan tidak ada *query* yang rumit dan kompleks. OLAP memiliki *query* yang berkebalikkan dengan OLTP hal ini karena data-data pada OLAP adalah data multidimensi sebagai data yang dibutuhkan analisis.

1. Manipulasi Data

OLTP pada proses *insert* dan *update* dapat dilakukan dengan cepat dan dapat dilakukan sendiri baik secara distribusi maupun online. OLAP proses *insert* dan *update* sangat jarang dilakukan, sebab OLAP memiliki data *historys* dan menggunakan *batch file* dan sistem secara periodik.

1. Aplikasi dan Layanan

OLPT lebih dominan pada aplikasi dan layanan yang berorientasi kepada operasional dan transaksi. OLAP lebih dominan pada aplikasi dan layanan yeng bertujuan untuk analisis data dan pengambilan keputusan dengan memanfaatkan data multidimensi.

## 2.13 *Data mart*

Menurut (Ghani, Jaber and Suryana, 2015) *Data mart is a storage of data that comes from various database or / and other sources of data that could support the business needs of the healthcare services. The data may be derived from various source of information such as dedicated database, other data warehouse and specific database. Data marts are able to provide a conceptual view of dataset elements through different data port.* Menurut (Ghani, Jaber and Suryana, 2015) *Data mart* sebuah penyimpanan data dari berbagai basis data berasal dari sumber data lain yang dapat mendukung kebutuhan bisnis. Data tersebut berasal dari berbagai macam sumber informasi seperti basis data khusus, gudang data lain, dan basis data tertentu.

Menurut (Golfarelli dan Rizzi) dalam (Supriyatna, 2016) *Data mart* dapat memberikan sudut pandang secara konseptual elemen dataset melalui port data yang beragam. *Data mart* adalah data yang berasal dari bagian *data warehouse* yang dibentuk dalam informasi yang relevan untuk melakukan sebuah keputusan, dalam bentuk kubus multi dimensional terkhusus di *query*-kan oleh *OLAP* dan *reporting front-ends*. *Data mart* mempunyai karakteristik yakni *data mart* tertuju pada kebutuhan user yang berhubungan dengan satu bagian departemen atau fungsi bisnis, *data mart* tidak berisi data operasional bersifat rinci, dan *data mart* sangat mudah dimengerti dan mudah digunakan karena data yang relatif lebih sedikit dibandingkan *data warehouse*.

Menurut (Permana R, 2013) *Data mart* melakukan proes data dengan menggunakan metode ETL (*Extract, Transform, dan Load*) dengan melakukan pemilihan dan penyeleksian data yang diproses dari sumber data, menstransformasikan data atau merubah data menjadi data yang diinginkan supaya dapat diproses pada *data mart* dan juga dapat melakukan *load* atau penyajian informasi kepada pengguna. Tujuan dari sistem *data mart* yaitu memudahkan pihak manajemen perusahaan dalam melakukan analisa pengambilan keputusan yang telah memiliki informasi akurat serta pengetahuan yang baik.

## 2.13.1 Perbedaan *Data mart* Dengan *Data warehouse*

Menurut (Standen) dalam (Pratama, 2018: 109) *Data mart* dan *data warehouse* memiliki perbedaan yanng signifikan, hal tersebut terlihat dari sudut pandang yang menyebabkan adanya 5 perbedaan yang didapatkan antara *data mart* dan *data warehouse* yakni :

1. *Data mart* dan *data warehouse* merupakan data yang berorientasi subjek, tetapi *data warehouse* meliputi seluruh area subjek sedangkan pada *data mart* hanya meliputi beberapa subjek berdasarkan kebutuhan dari sebuah organisasi.
2. Data yang dihasilkan oleh *data mart* merupakan data yang lebih spesifik yang merupakan data khusus berdasarkan kebutuhan sedangkan *data warehouse* data yang diberikan lebih detail dan sangat lengkap.
3. Dilihat dari sisi integrasi, *data mart* data yang terintegrasikan hanya spesifik data sesuai subjek atau dapat berdasarkan dari *data warehouse* sedangkan *data warehouse* dapat melakukan integrasi terhadap data secara luas dan menyeluruh dari berbagai sumber data. Data yang ada pada *data warehouse*
4. harus melakukan tahap proses ETL serta menyesuaikan dengan model data dan format baku untuk keselarasan yang ada pada data.
5. Berdasarkan data multidimensional, *data mart* lebih fokus pada model start *schema* yang merupakan bagian model data dimensional, sedangkan *data warehouse* tidak memiliki keharusan terhadap model dari *data multidimensional*.
6. Berdasarkan prinsip yang ada, *data mart* memiliki tujuan untuk membentuk informasi yang diberikan secara rinci dan spesifik, hal ini dikarenakan *data mart* bersifat *taktikal* dan praktis dalam penyedian data yang lebi spesifik, sedangkan *data warehouse* memudahkan proses analisis data dan pengambilan keputusan, dimana data yang dikumpulkan kemudian diintegrasikan berdasarkan kebutuhan untuk media informasi bagi pengguna. *Data warehouse* disebut sebagai konsep dala penyediaan serta dapat memberikan manfaat data dalam analisis.

## 2.14 *Microsoft SQL Server*

*SQL Server is a central part of the Microsoft data platform. SQL Server is an industry leader in operational database management systems (ODBMS). This documentation helps you install, configure, and use SQL Server on both Windows and Linux. The content includes end-to-end examples, code samples, and videos. For SQL Server language topic.* *SQL Server* merupakan komponen utama dari sebuah platform data Microsoft. *SQL Server* sebagai pemimpin industri untuk sistem manajemen basis data operasional (ODBMS). Hal ini memiliki tujuan untuk membantu menginstal, mengkonfigurasi, dan menggunakan *SQL Server* pada windows dan linux. *SQL Server* dapat diartikan sebagai produk *database* yang mudah digunakan serta dibuat. Kemudahan penggunaannya bisa dilihat mulai dari *interface* program *setup* yang memberikan petunjuk terhadap pengguna melalui proses instalasi.

## 2.15 *Business Intellegence*

Menurut (Sahu, Rajesh Kumar, 2012) dalam (Nur and Mukhlash, 2014) *Business intelligence* adalah seperangkat teori, metodologi, proses, aristektur, dan teknologi yang mengubah data mentah menjadi informasi yang bermakna serta berguna bertujuan tentang bisnis. BI menangani sejumlah informasi yang besar untuk membantu dalam mengidentifikasi dan mengembangkan sebuah peluang baru.

Menurut (Power, D.J, 2002) dalam (Nur and Mukhlash, 2014) *Business intelligence* adalah cara untuk mungumpulkan, menyimpan, mengorganisasikan, membentuk ulang kembali, meringkas data dan serta menyediakan informasi yang berupa aktifitas bisnis *internal* perusahaan, maupun data aktifitas bisnis *eksternal* perusahaan.

Menurut (Suryanto, Wahyu, dkk, 2010) dalam (Akbar *et al.*, 2017) *Business intelligence* adalah rangkaian kegiatan untuk memahami situasi bisnis dengan melakukan berbagai jenis analisis data yang dimiliki oleh organisasi serta data eksternal dari pihak ketiga untuk membantu menentukan strategi, keputusan bisnis yang taktis, dan operasional serta mengambil data yang diperlukan tindakan untuk mningkatkan kinerja bisnis. Terdapat empat komponen dasar dari *business intelligence* yang saling bersinambungan agar *business intelligence* dapat berfungsi berupa :

1. *Data warehouse*

*Data warehouse* merupakan sumber data dari business inteligence. *Data warehouse* yaitu koleksi data yang terorientasi pada subyek, tidak mengalami perubahan, dan memiliki kerentangan waktu yang cukup luas berfungsi sebagai pendukung pengambilan keputusan *management*.

1. *Business analytics*

*Business analytical* adalah sekelompok peralatan untuk manipulasi, menambahkan dan menganalisa data yang terdapat didalam *data warehouse.*

1. *Report* dan *Queries*

Bagian dari tahapan laporan baik secara statis (tidak berubah) ataupun dinamis sesuai dengan perubahan data dan setiap macam *query* yang ada seperti *drill down*, multidimensional *view*, pengelompokan dan sebagai beribut.

1. Data, text dan *web mining* dan peralatan matematika level atas dan statistik.

Langkah-langkah proses *business intelligence* menurut Ronald ada beberapa bagian dalam solusinya yaitu sebagai berikut :

1. Melakukan pengidentifikasian masalah bisnis yang perlu diselesaikan dengan gudang data dan menentukan data yang diperlukan untuk menyelesaikan masalah.
2. Identifikasi lokasi terhadap data-data yang diperlukan dan mengambilnya dari sumber penyimpnananya.
3. Mengubah data yang diperoleh dari beragam sumber kedalam sebuah data yang konsisten.
4. Melakukan pengambilan data yang telah diubah kedalam lokasi yang tersentralisasi.
5. Membangun sebuah gudang data dengan menggunakan data dalam lokasi yang tersentralisasi tersebut (*data warehouse*).

## 2.16 Pelayanan Jasa

Menurut (Alinaung, 2016) dalam (Marlius, 2018) Pelayanan merupakan aktivitas yang diberikan oleh satu pihak atau lebih kepada pihak lain yang memiliki hubungan dengan tujuan untuk memberikan kepuasan kedua belah pihak atas barang atau jasa yang diberikan. Pelayanan tersebut dapat meliputi, kecepatan melayani, kenyamanan yang diberikan, kemudahan lokasi, harga wajar dan bersaing.

Menurut (Lupiyohadi, 2009) dalam (Marlius, 2018) kualitas jasa didefinisikan seberapa jauh perbedaan antara kenyataan dan harapan pelanggan terhadap layanan yang mereka terima. Kualitas jasa dimulai dari kebutuhan pelangganan berakhir dengan kepuasan pelanggan serta persepsi positif terhadap kualitas jasa (Kodu, 2012) dalam (Marlius, 2018).

Menurut (Haryanto et al., 2012) dalam (Marlius, 2018) dalam kualitas pelayanan dijelaskan 5 kesenjangan yang menyebabkan adanya perbedaan dalam persepsi mengenai kualitas jasa yaitu meliputi kesenjangan persepsi manajemen, kesenjangan spesifikasi kualitas, kesenjangan penyampaian jasa, kesenjangan komunikasi pemasaran, dan kesenjangan dalam pelayanan yang dirasakan.

Menurut (Kereta, 2014) dalam (Marlius, 2018) terdapat determinan kualitas jasa berdasarkan urutan pentingnya yaitu terdiri dari:

1. Keandalan serta kemampuan untuk melaksanakan jasa yang dijanjikan dengan andal dan akurat.

2. Responsivitas dan ketersediaan membantu pelanggan juga memberikan layanan tepat waktu.

3. Jaminan, pengetahuan dan kesopanan karyawan serta kemampuan untuk memberikan keyakinan dan kepercayaan.

4. Empati, memerhatikan kondisi dan memberikan perhatian khusus terhadap pelanggan.

5. Mewujudkan, menampilakan penampilan fisik fasilitas, peralatan, personil, dan bahan komunikasi.

# BAB III

**METODOLOGI PENELITIAN**

# 3.1 Metodologi Penelitian

Metodologi adalah sebuah prosedur dalam penelitian yang wajib dilakukan penelti (Glesne, 2011). Metode adalah cara untuk mengumpulkan dan menganalisa data. Peneitian merupakan penyelidikan yang sistematis dapat berguna untuk meningkatkan pengetahuan dari suatu masalah yang terjadi. Dari penjelasan diatas dapat disimpulkan secara keseluruhan bahwa metodologi penelitian adalah sekumpulan langkah-langkah atau suatu prosedur yang digunakan peneliti untuk meyelesaikan permasalahan yang terjadi sehingga mendapatkan pengetahuan yang baru (Irawati, Sinaga, & Lubis, 2018)

# 3.2 Kerangka Penelitian

Kerangka penelitian sebuah urutan yang ada dalam suatu penelitian. Kerangka penelitian bertujuan untuk membuat perancangan agar tidak melewati batas dari pokok pembicaraan dan lebih mudah memahami isi penelitian. Maka dari itu dibentuklah kerangka penelitian untuk dijadikan sebagai pedoman yang jelas dan mudah untuk menyelesaikan permasalahan yang ada. Dari penelitian ini, maka dibentuk urutan langkah-langkah yang dapat dilihat dari gambar berikut :

**Penelitian Pendahuluan**

**Pengumpulan Data**

**Analisa Data**

**Identify and Conform The Deminision**

**Choose The *Grain***

**Choose The Process**

**Choose Fact**

**Store Precalculation in The Fact Table**

**Round Out The Deminisions Tables**

**Determine The Need To Track Slowly Changing Dimension**

**Dicide The Physical**

**Choose The Duration Of The *Database***

**Perancangan Model**

**Perancangan Sistem**

**Perancangan Interface**

**Implementasi**

**Pengujian**

**3.1 Kerangka Metodologi Penelitian**

# 3.3 Tahapan Penelitian

Penelitian (*research*) kegiatan ilmiah dalam menemukan, mengembangkan dan menguji sebuah kebenaran dalam pengetahuan atau masalah agar mencari pemecahan terhadap pengetahuan dan masalah tersebut. Secara keseluruhan data yang didapatkan dari penelitian tersebut dapat digunakan untuk memahami, memecahkan dan mengantisipasi suatu masalah yang ada. Tujuan dari peneltian yaitu untuk mencari sebuah kejelasan serta jawaban terhadap masalah dan memberikan alternatif kemungkinan yang dapat digunakan dalam pemecahan suatu masalah. Pemecahan masalah dan jawaban yang didapat terhadap masalah bersifat abstrak dan pada umumnya dalam penelitian dasar dan dalam spesifik seperti biasanya ditemui pada penelitian terapan.

Penelitian dapat dilakukan dengan menggunakan metode. Metode penelitian yaitu rangkaian cara atau kegiatan pelaksanaan penelitian berdasarkan langkah-langkah ilmiah yang disusun secara sistematis dan logis untuk dapat dijadikan pedoman yang jelas dan mudah dalam menyelesaikan masalah. Pada setiap tahapan penelitian saling berkaitan dan berhubungan dalam menentukan proses untuk menentukan tahap selanjutnya.

# 3.4 Penelitian Pendahuluan

Penelitian pendahuluan adalah langkah yang paling utama dalam melakukan penelitian dengan cara melakukan analisa terlebih dahulu masalah yang akan dikembangkan. Penelitian dilakukan di CV. RESTU GRAHA CIPTA, pada penelitian pendahuluan ini harus terlebih dahulu mengetahui sistem yang ada pada tempat penelitian tersebut dilakukan, agar dapat mencari tahu dan menganalisa semua permasalahan yang ada. Setelah melakukan analisa dan pengamatan, maka didapatkan masalah pada perusahaan tersebut yaitu rangkuman pelaporan penggunaan serta pemilihan tenaga ahli dalam sebuah proyek yang dijalankan. Oleh karena itu dalam pembuatan *data warehouse* ini dibutuhkan data personalia, tenaga ahli dan pembangunan *property* pada CV. RESTU GRAHA CIPTA . *Data warehouse* memberikan manfaat untuk mempermudah perusahaan dalam memberikan keputusan serta informasi untuk perkembangan perusahaan atau organisasi. Dan dapat memberikan grafik perkembangan berdasarkan data yang ada di perusahaan.

# 3.5 Pengumpulan Data

Pengumpulan data sangat diperlukan dalam peneltian ini, karena dalam perancangan *data warehouse* diperlukan data mentah dai perusahaan untuk dilakukannya penganalisaan. Ada beberapa cara untuk melakukan pengumpulan data yaitu meliputi :

1. Wawancara (*Interview*)

Pada tahap wawancara ini peneliti mengumpulkan data-data dengan cara melakukan pertemuan dengan pihak-pihak dari perusahaan. Dengan melakukan pertemuan tersebut, peneliti dapat melakukan sesi wawancara terhadap pihak perusahaan. Pada tahap ini dilakukan pertemuan dengan direktur perusahaan melakukan tanya jawab terkait tentang sistem pada perusahaan.

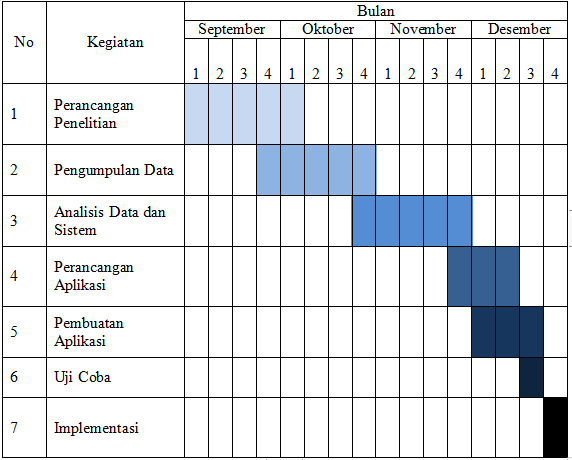
1. Pengumpulan Data Fisik Perusahaan

Setelah dilakukan wawancara, dan melakukan pengamatan untuk observasi, maka peneliti melakukan permintaan data fisik pada perusahaan untuk melakukan melakukan riset apakah data yang didapatkan sesuai dan dapat digunakan dalam melakukan penelitian.

# 3.6 Waktu Penelitian

Waktu penelitian dimulai sejak bulan September 2019

**Tabel 3.1 Jadwal Penelitian**



Keterangan Waktu Jadwal Penelitian :

Dilaksanakan pada tanggal 4 September - 1 Oktober 2019

Dilaksanakan pada tanggal 23 September - 29 Oktober 2019

Dilaksanakan pada tanggal 24 Oktober - 30 November 2019

Estimasi implementasi berkisar antara 24 Desember – 7 Desember 2019

Estimasi implementasi berkisar antara 7 Desember – 20 Desember 2019

Estimasi implementasi berkisar antara 20 Desember – 21 Desember 2019

Estimasi implementasi berkisar antara 21 November – 23 Desember 2019

# 3.7 Analisa

Menurut Kimball dalam (Connolly and Begg, 2005) ada 9 langkah dalam membangun sebuah *data warehouse* yang dikenal dengan sebutan *nine-step design methodology.* Berikut penjelasan dari tahapan yang akan digunakan dalam analisa penelitian ini :

1. ***Choosing The Proces* (Memilih Proses)**

Pemilihan proses merujuk pada subjek utama yang merupakan kegiatan bisnis perusahaan CV. Restu Graha Cipta merupakan perusahaan yang bergerak dibidang konsultasi pembangunan, lebih memfokuskan terhadap tenaga ahli yang akan digunakan serta *property* yang akan dibangun. Perusahaan ini bergerak dalam penjualan jasa kepada perusahaan lain. Data yang ada meliputi dari : Data tenaga ahli, data bidang perencanaan atau *property*, dan data fasilitas.

1. ***Choosing The Grain* (Memilih *Grain*)**

Proses pemilihan *grain* merupakan proses pemilihan dimensi yang akan dimasukkan untuk ketabel fakta yang akan dianalisis. Memilih *grain* untuk menentukan yang akan direpresentasikan oleh *record* dan tabel fakta. *Grain* pada CV. Restu Graha Cipta digunakan untuk merancang *data warehouse* adalah penjualan. Analisa pada penjualan CV. Restu Graha Cipta yaitu jasa yang paling banyak dipilih untuk melakukan proyek pembangunan *property*.

1. ***Identifying and Conforming The Dimensions* (Mendefinisikan dan Menyesuaikan Dimensi)**

Mendefinisikan dan menyesuaikan dimensi merupakan sekumpulan dari sudut pandang dimensi yang sangat penting untuk mendefinisikan fakta yang terdapat pada tabel fakta.

1. ***Choosing The Fact* (Memilih Tabel Fakta)**

Proses dalam memilih tabel fakta yang akan digunakan dalam *data warehouse*. Masing-masing dari fakta memiliki data yang dapat dihitung. Data tersebut ditampilkan dalam bentuk laporan, grafik, atau diagram lainnya. Berikut merupakan fakta-fakta yang akan ditampilkan di dalam *data warehouse*:

1. Jasa Personalia / Tenaga Kerja meliputi id\_tenagaahli, kd\_statustenagaahli, kd\_pendidikan, kd\_waktu, kd\_daerah, rata\_rataumur, dan total\_statustenagaahli
2. Jasa Konsultasi Perencanaan/ instansi meliputi kd\_instansi, kd\_subbidang, kd\_daerah, id\_pemberijasa, id\_tenagaahli, kd\_waktu, Nilaikontrak, total\_instansi.
3. Fasilitas meliputi kd\_namafasilitas, kd\_jenisfasilitas, kd\_merekfasilitas, kd\_ketfasilitas, kd\_waktu, kd\_lokasifasilitas, kd\_kondisifasilitas, kd\_daerah dan total\_fasilitas.
4. ***Storing Pre-Calculations In The Fact Table* ( Menyimpan Pre-Kalkulasi dalam Tabel Fakta)**

Menyimpan Pre-kalkulasi dalam tabel fakta merupakan proses ketika tabel fakta telah dipilih, maka tabel fakta tersebut diuji apakah fakta yang dapat menggunakan pre-kalkulasi. Kemudian lakukan penyimpanan pada tabel fakta.

1. ***Rounding Out The Dimension Tables* ( Melengkapi Tabel Dimensi)**

Menambahkan deskripsi teks pada dimensi. Deskripsi yang dibuat harus bersifat *intuitif* dan dapat dipahami oleh *user*. Dimensi *field* deskripsi waktu terdiri dari tahun, bulan, minggu. Hal ini dapat dapat merincikan bahwa laporan dapat dilihat berdasarkan per tahun, per bulan, per tiga bulan, per minggu bahkan per hari. Fasilitas dapat dilihat berdasarkan item, transportasi dapat dilihat berdasarkan item, laporan tenaga ahli dapat dilihat berdasarkan tenaga ahli, laporan bidang pembangunan atau *property* dapat dilihat berdasarkan *property*.

1. ***Choose The Duration Of The Database* ( Memilih Durasi Dari Basis Data)**

Dalam pemilihan durasi, durasi yang ditarget dalam penelitian ini berkisar 5 tahun untuk sebuah *data warehouse*. Maka data yang dimiliki memiliki rentan waktu selama 5 tahun. Hal ini bertujuan untuk dapat memberikan hasil serta perkembangan berdasarkan data yang dimiliki.

1. ***Determine The Need To Track Slowly Changing Dimension* (Melacak Perubahan Dimensi Secara Bertahan)**

Melacak perubahan dimensi secara bertahan merupakan prose tahapan yang terjadinya perubahan atribut pada dimensi yang akan menyebabkan penambahan *record* dimensi baru, membuat atribut alternatif baru untuk menampung nilai baru dan membuat kembali atribut yang berubah. Hal ini disebabkan untuk menjaga data yang lama tetap ada dan dapat mengetahui peruahan dimensi yang terjadi pada data lama dan data baru. Contohnya terjadinya perubahan tanggal lahir dan nama tenaga ahli yang mengakibatkan penambahan *record* baru pada dimensi dengan tetap menyimpan *record* lama.

1. ***Dicide The Physical Design***

Proses yang merupakan untuk perancangan fisik dari *data warehouse.* Perancangan yang ada pada *data warehouse* ini menggunakan *snowflakes schema*. Karena *snowflakes schema* memiliki kelebihan meliputi :

1. Skema snowflake menggunakan ruang penyimpanan yang lebih kecil dibandingkan ruang penyimpanan pada skema bintang.
2. Penyimpanan data pada skema snowflake lebih terorganisir dan lebih rapi dibandingkan dengan skema bintang.
3. Proses *querry* lebih cepat pada proses OLAP (*Online Analytical Processing*)

# 3.8 Perancangan

Perancangan merupakan proses tahapan yang dilakukan untuk penentuan tabel dimensi dan tabel fakta dari data mentah yang didapatkan untuk merancang *data warehouse*. Pada masing-masing tabel dimensi memiliki primary key yang akan masuk kedalam tabel fakta. Penggunaan *primary key* untuk mewakili satu jenis data. Pada tabel fakta berisi dari *primary key* masing-masing tabel dimensi dan measure yang diinginkan. Setelah tabel dimensi dan fakta dibangun, maka kedua tabel tersebut dihubungkan dengan meggunakan *snowflakes schema*. Pada skema bintang merupakan tabel fakta yang dikelilingi oleh tabel dimensi.

Model perancangan yang digunakan pada penelitian ini yaitu *snowflakes schema*, mendesain proses ETL (*Extraction, Transformation, and Loading*) , analisa ERD dan menganalisis meta data yang digunakan dalam *data warehouse*. Alasan penggunaan dengan menggunakan *snowflakes schema* yaitu ukuran penyimpanan kecil didalam penyimpanan data yang menyebabkan terjadinya penghematan ruang penyimpanan data dan struktur yang normal lebih mudah di-update dan di-maintenance. Pada *snowflakes schema* seluruh tabel telah dinormalisasi, proses ini disebut dengan *snowflaking*. Sebagai kebutuhan untuk pengambilan keputusan tingkat tinggi, dengan tipe ini seluruh struktur dapat digunakan sepenuhnya.

# Implementasi

Implementasi pada penelitian ini dapat dilakukan setelah perancangan selesai dilakukan. Pada tahap ini akan dilakukan pengolahan data mentah dari perusahaan untuk dapat menjadikan syarat membangun *database* untuk *data warehouse* didalam SQL Server 2015 dan diterapkan kedalam aplikasi Web untuk memanggil *key* yang telah diolah sebelumnya. Hal ini bertujuan agar admin dapat melakukan pengimplementasian dengan mudah dalam pengolahan *data warehouse*.

Proses dari pengujian memiliki tujuan untuk melihat apakah *data warehouse* yang dibuat telah sesuai dengan penelitian dan telah dapat menyelesaikan masalah yang ada pada Cv. Restu Graha Cipta dan memberikan solusi yang didapat setelah dilakukannya pengolahan. Pengujian dilakukan dengan menggunakan pengujian *black box* yaitu pengujian yang memperhatikan struktur logika internal perangkat lunak. Pengujian *black box* digunakan untuk mengetahui perangkat lunak berfungsi dengan benar dan merupakan metode perancangan data uji yang didasarkan pada spesifikasi perangkat lunak.

# 3.10 Pengujian

Pengujian black box memiliki tujuaan dalam menemukan kesalahan yang tersusun dalam kategori :

1. Fungsi-fungsi yang tidak benar dan hilang.
2. Kesalahan interface.
3. Kesalahan dalam struktur data atau akses database ekternal.
4. Kesalahan dalam kinerja.
5. Inisialisasi dan kesalahan terminasi.

# BAB IV

**ANALISA DAN PERANCANGAN SISTEM**

# 4.1 Analisa

# Analisa Data

Dalam menganalisa kebutuhan ada beberapa hal yang perlu diperhatikan dalam penganalisaan tersebut yaitu dapat dilihat sebagai berikut :

1. Mendefinisikan Kebutuhan Fungsional *Data warehouse*

Dalam mendefinisikan kebutuhan fungsional harus melakukan diskusi dengan pengguna yang akan menggunakan *data warehouse* tersebut. Dari hasil observasi dan wawancara dapat disimpulkan CV. Restu Graha Cipta memerlukan sebuah *data warehouse*. Berikut fungsional dari *data warehouse* yang akan menjawab kebutuhan informasi yang diminta.

1. *Data warehouse* harus dapat melakukan analisa terhadap data tenaga ahli.
2. *Data warehouse* dapat melakukan analisa terhadap data konsultan perencanaan.
3. *Data warehouse* harus dapat melakukan analisa terhadap data fasilitas.
4. Studi Kelayakan

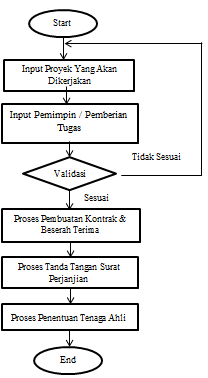
Studi kelayakan data merupakan proses yang digunakan dalam memeriksa sumber sistem, untuk memahami data dengan membuat daftar data resiko utama dan melakukan verifikasi kepada pengguna. Adapun daftar resiko dalam perancangan *data warehouse* antara lain :

1. Tidak dapat melakukan *ekstraksi* hanya dalam waktu 1 jam
2. Kebutuhan fungsional tidak terpenuhi jika tidak adanya data dalam sistem sumber.
3. Data yang diinginkan sudah didapat, tetapi apabila data masih belum terstruktur maka harus direstrukturisasi dan dibersihkan terlebih dahulu.
4. Kapasitas data tambahan setiap hari membuat sistem sumber menjadi lebih lambat.
5. *Flowchart* Pada Data Perusahaan CV. Restu Graha Cipta

Proses ini merupakan alur kerja data pada perusahaan. Hal ini bertujuan agar data yang ada pada perusahaan memiliki tahapan yang terorganisir dengan baik

1. Flowchart Konsultaan Instansi

Flowchart untuk menentukan proses dalam melakukan konsultan instansi pada perusahaan CV. Restu Graha Cipta, berikut ini merupakan flowchart konsultan instansi gambar dibawah ini :



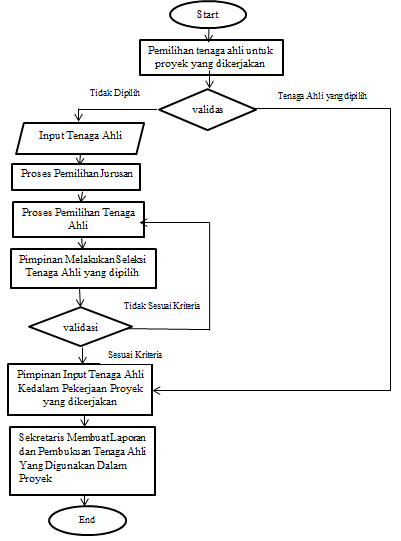
**Sumber : Wawancara Pimpinan CV. RESTU GRAHA CIPTA**

**Gambar 4.1 Flowchart Proses Konsultan Instansi**

Pada gambar diatas proses konsultan instansi. Dimulai dari menentukan proyek yang akan dikerjakan terlebih dahulu, menentukan instansi yang akan dikerjakan dan melakukan pemilihan pemimpin dalam proyek instansi, apabila pemimpin menolak karena tidak sesuai dengan instansi yang diinginkan dilakukan kembali proses penentuan instansi yang baru, dan apabila pemimpin melakukan proses pembuatan kontrak serta beserah terima antara pemimpin dan pihak perusahaan untuk menentukan waktu kontrak. Kedua pihak melakukan proses tanda tangan yang menandakan proyek instansi tersebut dapat dikerjakan, Setelah itu pihak perusahaan memilih tenaga kerja ahli untuk mengkoordiantor proyek instansi yang akan dikerjakan.

1. Flowchart Tenaga Ahli

Flowchart untuk menentukan proses dalam melakukan pada tenaga ahli perusahaan CV. Restu Graha Cipta, berikut ini merupakan flowchart tenaga ahli gambar dibawah ini :



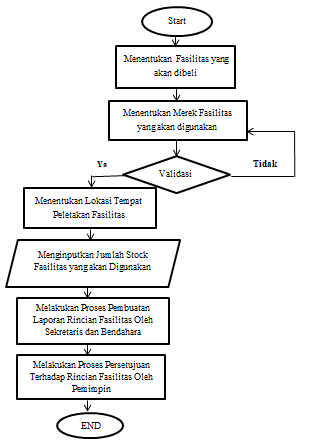
**Sumber : Wawancara Pimpinan CV. RESTU GRAHA CIPTA**

**Gambar 4.2 Flowchart Proses Pemilihan Tenaga Ahli**

Pada Gambar 4.2 proses dalam pemilihan tenaga ahli oleh perusahaan dalam mengkoordinator poyek instansi yang akan dikerjakan. Perusahaan melakukan pemilihan tenaga ahli dengan menginputkan data tenaga ahl dan jurusan yang diinginkani agar proyek dapat dipegang oleh tenaga ahli sesuai. Dan jika sesuai maka pimpinan akan menginputkan tenaga ahli yang dipilih kedalam data proyek instansi yang akan dikerjakan oleh tenaga ahli jika tidak maka terjadinya proses penginputan data tenaga ahli kembali. Sekretaris membuat data laporan dan pembukuan tenaga ahli yang digunakan dalam proyek instansi yang dikerjakan dan proses selesai.

1. Flowchart Fasilitas

Flowchart untuk menentukan proses dalam melakukan pemilihan fasilitas perusahaan CV. Restu Graha Cipta, berikut ini merupakan flowchart fasilitas gambar dibawah ini :



**Sumber : Wawancara Pimpinan CV. RESTU GRAHA CIPTA**

**Gambar 4.3 Flowchart Proses Pemilihan Fasilitas**

Pada gambar 4.3 proses dalam pemilihan fasilitas yang akan digunakan diperusahaan tersebut. Menentukan fasilitas yang akan digunakan perusahaan, kemudian menentukan merek serta spesifikasi yang sesuai untuk kebutuhan perusahaan, jita tidak sesuai maka dilakukan pemilihan merek serta spesifikasi fasilitas kembali dan apabila sesuai maka perusahaan menentukan tempat dalam meletekkan fasilitas yang akan digunakan, selanjutnya menentukan jumlah stock fasilitas yang dibutuhkan setelah itu melakuka proses pembuatan laporan rincian fasilitas, hal ini dilakukan oleh sekretaris serta bendahara perusahaan tersebut. Hasil dari laporan tersebut diberikan kepada pimpinan sebagai bentuk persetujuan pemimpin untuk melakukan pengeluaran keuangan terhadap fasilitas yang dibeli.

1. Permasalahan Penelitian

Berdasarkan penelitian pada CV. Restu Graha Cipta terhadap pelaksaan sistem dan prosedur yang berjalan saat ini, terdapat beberapa masalah yang dihadapi oleh pihak perusahaan yaitu :

1. Belum adanya sistem yang digunakan sebagai sarana penunjang dalam pengambilan keputusan mengenai permasalahan yang terjadi di perusahaan.
2. Pelaporan hasil dari data yang diperoleh kurang optimal terhadap pimpinan perusahaan.
3. Alternatif Pemecahan Masalah

Berdasarkan masalah yang telah dijelaskan diatas, maka solusi masalah yang dihadapi yaitu membangun *data warehouse* pada perusahaan tanpa menggangu sistem yang sedang berjalan. *Data warehouse* ini sangat diperlukan karena data yang akan disatukan berasal dari seluruh data yang ada di perusahaan. Khususnya data tenaga ahli, data konsultan perencanaan, dan data fasilitas. Fungsi dari *data warehouse* ini untuk melihat peningkatan pemasukan proyek yang ditangani setiap tahunnya dan melihat data peningkatan penggunaan tenaga ahli yang digunakan.

1. Analisis Kebutuhan Data dan Informasi

Dalam penelitian untuk mengetahui data dan informasi yang diperlukan oleh pihak perusahaan yang terkait, maka peneliti melakukan pengumpulan data dengan metode wawancara dan pengumpulan data fisik perusahaan yang diberikan langsung oleh pimpinan perusahaan.

Informasi yang diproleh dari hasil wawancara tersebut yaitu pimpinan perusahaan memerlukan laporan yang jelas dan cepat dalam melakukan pengambilan keputusan kedepannya. Informasi yang diterima antara lain :

1. Informasi penggunaan tenaga ahli tetap dan non tetap yang meliputi jumlah tenaga ahli tetap dan jumlah tenaga ahli non tetap.
2. Informasi pemilihan proyek yang dikerjakan meliputi jumlah keseluruhan jumlah proyek yang dikerjakan.
3. Informasi target terhadap proyek yang dikerjakan meliputi jumlah proyek yang telah tercapai dan jumlah proyek yang tidak tercapai.
4. Informasi pemilihan pemimpin dalam pemberian tugas dan jasa meliputi jumlah pemberian jasa secara keseluruhan yang digunakan selama setahun.
5. Informasi fasilitas yang digunakan meliputi jumlah fasilitas yang secara keseluruhan yang digunakan.
6. Laporan hasil dari *data warehouse* antara lain laporan tenaga ahli, laporan fasilitas, laporan konsultan instansi.

# Analisa Sistem

Pada proses analisis sistem membahas mengenai gambaran kegiatan sistem pada CV. Restu Graha Cipta secara keseluruhan. Proses analisis ini memiliki tujuan untuk memudahkan *analyst* dalam membuat suatu sistem yang akan berjalan serta perancangan *data warehouse* bagi perusahaan yang akan menghasilkan suatu informasi yang berguna untuk perusahaan tersebut. Untuk memperoleh data atau informasi pada penelitian ini sebaiknya melakukan kegiatan pengumpulan data fisik yang berguna sebagai penunjang untuk melakukan objek penelitian.

Pada analisa penelitian *data warehouse* ini, penulis menggunakan metodologi perancangan *data warehouse* yang dikemukakan oleh Kimball berdasarkan (Connoly and Begg) yaitu dikenal dengan *Nine-Step Methodology terdiri dari choosing the process, choosing the grain , indetifying and conforming the dimensions , choosing the fact, storing pre-calculations in the fact table, rounding out the dimensions, choosing the durations of the database, determine the need to track slowly changing dimensions, dicide the physical design.*

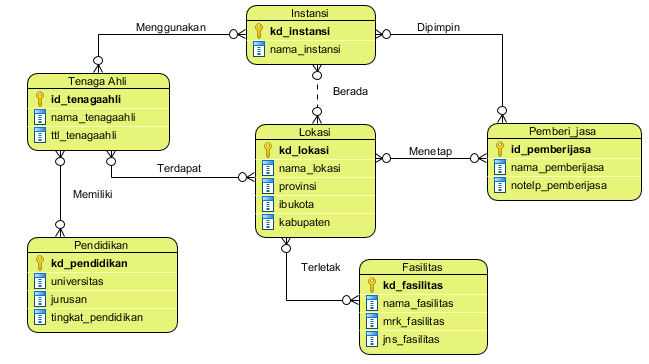
# Analisa Proses

# *Nine-Step Design Methodology Data Warehouse*

Perancangan *data warehouse* ini mengunakan metodologi perancangan *data warehouse* yang dikemukan oleh kimball *yaitu nine-step design methodology* berikut merupakan tahapan-tahapannya :

1. ***Choosing The Process***

Pemilihan proses atau dikenal dengan *choosing proses* yaitu penentuan subjek dari masalah yang sedang dihadapi. Berdasarkan dari hasil penelitian observasi dan wawancara yang dilakukan ada beberapa proses terpenting dalam kegiatan operasional yang terkait dengan perusahaan ini. Serta pembuatan ERD (*Entity Relationship Diagram*) pada data yang telah didapatkan. Berikut ini ERD data CV. Restu Graha Cipta sebagai berikut :



**Gambar 4.4 Entity Relationship Diagram CV. Restu Graha Cipta**

ERD pada gambar diatas terdiri dari 8 entitas yang masing-masing merupakan dari *Lokasi, Instansi, Pemberi Jasa, Tenaga Ahli, Fasilitas, Pendidikan dan Kondisi*. Entitas tersebut memiliki hubungan atau relasi yang terjadi sebagaiberikut :

1. Banyak sampai banyak *Tenaga ahli* terdapat banyak sampai banyak *lokasi*.
2. Banyak sampai banyak *Tenaga Ahli* digunakan banyak sampai banyak *Instansi*.
3. Banyak sampai banyak *Instansi* dipimpin banyak sampai banyak *Pemberian Jasa*.
4. Banya sampai banyak *Tenaga Ahli* memiliki banyak sampai banyak *Pedidikan*.
5. Banyak sampai banyak *pemberi jasa* menetap banyak sampai banyak *lokasi*.
6. Banyak sampai banyak *fasilitas* terletak banyak sampai banyak *lokasi*.

Dari data OLAP yang telah diperoleh maka akan dirancangnya *entity relationship diagram* untuk menjadi *fact table*. ERD tersebut dibuat setelah dilakukan penganalisaan database OLAP yang sedang berjalan dan yang diperoleh dari CV. Restu Graha Cipta yang terdiri dari 3 tabel fakta. Masing-masing fakta tergabung kedalam entitas yang terdapat pada ERD untuk direlasikan kepada entitas yang lainnya.

1. ***Choosing The Grain***

*Grain* adalah data dari calon tabel fakta yang akan dianalisa. Dalam melakukan pemilihan *grain*, dapat diperkirakan yang direpresentasikan pada *record* tabel fakta. Setelah menentukan *grain* tabel fakta, maka untuk dimensi pada tiap tabel fakta dapat diindentifikasi. Grain yang terdapat pada perancangan *data warehouse* tabel fakta meliputi :

1. Laporan Fasilitas

Laporan fasilitas yang dimiliki oleh perusahaan meliputi data yang dianalisis yaitu nama fasilitas, jenis fasilitas, merek fasilitas, keterangan fasilitas, kondisi fasilitas, waktu, total fasilitas, daerah, dan jumlah fasilitas.

1. Laporan Tenaga Ahli

Laporan Tenaga Ahli data yang meliputi, data yang dianalisis meliputi nama tenaga ahli, status tenaga ahli, tingkat pendidikan, waktu, daerah, rata-rata umur, jumlah tenaga ahli, total berdasarkan status.

1. Laporan Konsultan Perencanaan

Dalam laporan konsultan perencanaan, data yang dianalisis meliputi data nama instansi, nama tenaga ahli yang digunakan dalam proyek instansi, sub bidang yang dimiliki oleh tenaga ahli, pemberi jasa, total instansi, dan nilai kontrak instansi yang dikerjakan.

1. ***Identifying and Conforming The Dimensions***

Identifikasi dan penyesuaian dimensi atau dikenal *Identifying and Conforming The Dimensions* merupakan tahapan atau proses yang dilakukan untuk penyesuaian dimensi dengan menggunakan fakta yang ada. Tahap yang melibatkan sudut pandang dalam mendefinisikan fakta yang akan ada dalam tabel fakta. Berikut ini proses identifikasi dan konfirmasi dimensi sebagai berikut :

**Tabel 4.1 Rancangan Matriks Proses dan Dimensi**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Dimensi/Grain** | **Nama Fasilitas** | **Jenis Fasilitas** | **Merek Fasilias** | **Keterangan Fasilitas** | **Kondisi Fasilitas** | **Waktu** | **Tenaga Ahli** | **Tingkat Pendidikan** | **Status Tenaga Ahli** | **Jurusan** | **Universitas** | **Instansi** | **Instansi Sub Bidang** | **Pemberi Jasa** | **Daerah** | **Kabupaten** | **Provinsi** | **Ibu Kota** |
| Jenis Fasilitas yang sering mengalami kerusakan | √ | √ |  |  | √ |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Fasilitas yang layak untuk digunakan | √ |  |  |  | √ |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Merek fasilitas yang masih berkualitas bagus | √ |  | √ |  | √ |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Fasilitas yang merupakan milik pribadi | √ |  |  | √ |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Fasilitas yang merupakan sewa | √ |  |  | √ |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Jangka waktu fasilitas yang disewa | √ | √ | √ | √ |  | √ |  |  |  |  |  |  |  |  |  |  |  |  |
| Merek fasilitas yang disewa | √ |  | √ | √ |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Fasilitas yang disewa oleh perusahaan | √ | √ | √ | √ |  | √ |  |  |  |  |  |  |  |  |  |  |  |  |
| Jumlah Fasilitas yang digunakan oleh perusahaan | √ | √ | √ | √ | √ |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Tenaga ahli yang sering digunakan |  |  |  |  |  |  | √ | √ | √ |  |  |  |  |  |  |  |  |  |
| Alamat pemberian jasa instansi |  |  |  |  |  |  |  |  |  |  |  |  |  | √ |  |  |  |  |
| **Dimensi/Grain** | **Nama Fasilitas** | **Jenis Fasilitas** | **Merek Fasilias** | **Keterangan Fasilitas** | **Kondisi Fasilitas** | **Waktu** | **Nama Tenaga Ahli** | **Tingkat Pendidikan** | **Status Tenaga Ahli** | **Jurusan TenagaAhli** | **Universitas** | **Nama Instansi** | **Instansi Sub Bidang** | **Nama Pemberi Jasa** | **Daerah** | **Kabupaten** | **Provinsi** | **Ibu Kota** |
| Tenaga ahli yang jarang digunakan |  |  |  |  |  |  | √ | √ | √ |  |  |  |  |  |  |  |  |  |
| Tingkat pendidikan tenaga ahli |  |  |  |  |  |  | √ | √ |  |  |  |  |  |  |  |  |  |  |
| Jurusan akademik tenaga ahli |  |  |  |  |  |  | √ | √ | √ | √ |  |  |  |  |  |  |  |  |
| Universitas Tamatan Tenaga Ahli |  |  |  |  |  |  | √ | √ | √ | √ | √ |  |  |  |  |  |  |  |
| Status Tenaga Ahli dalam perusahaan |  |  |  |  |  |  | √ | √ | √ |  |  |  |  |  |  |  |  |  |
| Pengalaman tenaga ahli bekerja |  |  |  |  |  |  | √ | √ | √ |  |  |  |  |  |  |  |  |  |
| Tenaga ahli yang bekerja mengalami peningkatan tiap tahunnya |  |  |  |  |  |  | √ | √ | √ | √ | √ |  |  |  |  |  |  |  |
| Jumlah tenaga ahli yang masih bekerja |  |  |  |  |  | √ | √ | √ | √ | √ | √ |  |  |  |  |  |  |  |
| Umur tenaga ahli yang bekerja diperusahaan |  |  |  |  |  | √ | √ |  |  |  |  |  |  |  |  |  |  |  |
| Tenaga ahli yang digunakan dalam instansi untuk dikerjakan |  |  |  |  |  | √ | √ |  | √ |  |  | √ |  |  |  |  |  |  |
| Nama pemberian jasa instansi yang dikerjakan |  |  |  |  |  |  |  |  |  |  |  | √ | √ | √ |  |  |  |  |
| Jumlah Proyek instansi yang dikerjakan oleh pemberian jasa |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Dimensi/Grain** | **Nama Fasilitas** | **Jenis Fasilitas** | **Merek Fasilias** | **Keterangan Fasilitas** | **Kondisi Fasilitas** | **Waktu** | **Nama Tenaga Ahli** | **Tingkat Pendidikan** | **Status Tenaga Ahli** | **Jurusan TenagaAhli** | **Universitas Tenaga Ahli** | **Nama Instansi** | **Instansi Sub Bidang** | **Nama Pemberi Jasa** | **Daerah** | **Kabupaten** | **Provinsi** | **Ibu Kota** |
| Sub bidang tenaga ahli digunakan pada instansi |  |  |  |  |  |  |  |  |  |  |  | √ | √ | √ |  |  |  |  |
| Lokasi instansi yang dikerjakan |  |  |  |  |  |  |  |  |  |  |  | √ |  |  | √ | √ | √ | √ |
| Waktu instansi dikerjakan tepat waktu |  |  |  |  |  | √ |  |  |  |  |  | √ | √ |  |  |  |  |  |
| Nilai Kontrak instansi yang dikerjakan |  |  |  |  |  | √ |  |  |  |  |  | √ | √ |  |  |  |  |  |
| Instansi yang dikerjakan bernilai 100 juta keatas |  |  |  |  |  |  |  |  |  |  |  | √ | √ |  |  |  |  |  |
| Instansi yang dikerjakan bernilai dibawah 100 juta |  |  |  |  |  |  |  |  |  |  |  | √ | √ |  |  |  |  |  |
| Instansi yang dikerjakan dengan waktu pertahun |  |  |  |  |  | √ |  |  |  |  |  | √ | √ |  |  |  |  |  |
| Instansi yang dikerjakan dilokasi yang sama |  |  |  |  |  |  |  |  |  |  |  | √ | √ |  | √ | √ | √ | √ |
| Waktu penyelesaian instansi yang tidak sesuai dengan jadwal yang ditentukan |  |  |  |  |  | √ |  |  |  |  |  | √ | √ |  |  |  |  |  |
| Tenaga ahli yang digunakan instansi masih tetap sama |  |  |  |  |  |  | √ |  |  |  |  | √ | √ |  |  |  |  |  |
| Jumlah proyek isntansi yang dikerjakan pertahun |  |  |  |  |  |  |  |  |  |  |  | √ | √ | √ |  |  |  |  |

Pada tabel 4.1 diatas menjelaskan tentang identifikasi dan penyesuaian dimensi pada data yang diperoleh dari perusahaan. Identifikasi yang dilakukan untuk mengetahui fungsi dimensi dari informasi yang didapatkan. Seluruh dimensi dideskripsikan sebanyak mungkin untuk mengetahui fakta yang ada pada dimensi-dimensi yang telah ada. Ada 18 dimensi untuk 3 tabel fakta yang dimiliki, yang terdiri dari fakta fasilitas, fakta tenaga ahli dan fakta konsultan perencanaan. Pada dimensi memiliki daerah memiliki foreign key yang merupakan turunan dari dimensi sebelumnya.

1. ***Choosing The Fact***

*Choosing the fact* atau memilih tabel fakta yang digunakan dalam *data warehouse*. Pada fakta memiliki data yang dapat dihitung yang akan ditampilkan dalam bentuk laporan, grafik, atau diagram. Pemilihan fakta dilakukan berdasarkan kebutuhan. Fakta terbentuk dari atribut dimensi dari data measure. Fakta yang terbentuk dari peracangan *data warehouse* yaitu:

1. Fasilitas meliputi kd\_fasilitas, kd\_jnsfasilitas, kd\_merekfasilitas, kd\_ketfasilitas, kd\_kondisifasilitas, kd\_waktu, kd\_daerah, total\_fasilitas, jumlah\_fasilitas.
2. Tenaga Ahli meliputi id\_tenaga ahli, kd\_status, kd\_universitas, kd\_jurusan, kd\_tingkatpendidikan, kd\_waktu, kd\_daerah, rata-rataumur, total\_status.
3. Konsultan Perencanaan meliputi kd\_instansi, kd\_subbidang, id\_tenagaahli, id\_pemberianjasa, kd\_waktu, kd\_daerah, total\_instansi, nilai\_kontrak, total\_tenagaahli.

**Tabel 4.2 Tabel Fakta Fasilitas**

|  |  |  |
| --- | --- | --- |
| **Nama Kolom** | **Tipe Data** | **Panjang** |
| Kd\_namafasilitas | Number | 5 |
| Kd\_jnsfasilitas | Number | 5 |
| Kd\_merekfasilitas | Number | 5 |
| Kd\_ketfasilitas | Number | 5 |
| Kd\_kondisifasilitas | Number | 5 |
| Kd\_waktu | Number | 5 |
| Kd\_daerah | Number | 5 |
| Kd\_kabupaten | Number | 5 |
| Kd\_provinsi | Number | 5 |
| Kd\_ibukota | Number | 5 |
| Total\_fasilitas | Number | 5 |

Pada tabel 4.2 merupakan tabel fakta fasilitas yang terdiri dari beberapa *primary key* dari beberapa dimensi yang saling terhubung satu sama lain pada tabel fakta. *Primary key* yang terdapat pada tabel fakta fasilitas terdiri dari : kd\_fasilitas, kd\_jnsfasilitas, kd\_merekfasilitas, kd\_ketfasilitas, kd\_kondisifasilitas, kd\_waktu, kd\_daerah, kd\_kabupaten, kd\_provinsi, dan kd\_ibukota. *Measure* yang ada pada tabel fakta fasilitas yaitu total\_fasilitas*. Primary key* pada fakta dapat digunakan untuk mengambil suatu keputusan dan dapat diolah berdasarkan keinginan.

**Tabel 4.3 Tabel Fakta Tenaga Ahli**

|  |  |  |
| --- | --- | --- |
| **Nama Kolom** | **Tipe Data** | **Panjang** |
| Id\_tenagaahli | Number | 5 |
| Kd\_status | Number | 5 |
| Kd\_tingkatpendidikan | Number | 5 |
| Kd\_waktu | Number | 5 |
| Kd\_daerah | Number | 5 |
| Kd\_kabupaten | Number | 5 |
| Kd\_provinsi | Number | 5 |
| Kd\_ibukota | Number | 5 |
| Kd\_jurusan | Number | 5 |
| Kd\_universitas | Number | 5 |
| Rata\_umur | Number | 5 |
| Total\_status | Number | 5 |

Pada tabel 4.3 merupakan tabel fakta tenaga ahli yang terdiri dari beberapa *primary key* pada dimensi yang saling terhubung satu sama lain ke tabel fakta dan tipe data *numeric* yang ada tabel fakta. *Primary key* dapat diolah sesuai keinginan, yang terdapat pada tabel fakta tenaga ahli terdiri dari : id\_tenaga ahli, kd\_status, kd\_tingkatpendidikan, kd\_jurusan, kd\_universitas, kd\_waktu, kd\_daerah, kd\_kabupaten, kd\_provinsi, dan kd\_ibukota. *Measure* yang ada pada fakta tenaga ahli terdiri dari, rata\_umur dan total\_status, dan total\_tingkatpendidikan.

**Tabel 4.4 Tabel Fakta Konsultan Instansi**

|  |  |  |
| --- | --- | --- |
| **Nama Kolom** | **Tipe Data** | **Panjang** |
| Kd\_instansi | Number | 5 |
| Kd\_subbidang | Number | 5 |
| Id\_tenagaahli | Number | 5 |
| Id\_pemberijasa | Number | 5 |
| Kd\_waktu | DateTime | - |
| Kd\_daerah | Number | 5 |
| Kd\_kabupaten | Number | 5 |
| Kd\_provinsi | Number | 5 |
| Kd\_ibukota | Number | 5 |
| Total\_Instansi | Number | 5 |
| Total\_nilaiinstansi | Number | 5 |

Pada tabel 4.4 merupakan tabel fakta konsultan perencanaan yang terdiri dari beberapa *primary key* pada dimensi yang saling terhubung satu sama lain ke tabel fakta dan tipe data *numeric* yang ada tabel fakta. *Primary key* yang terdapat pada tabel fakta tenaga ahli terdiri dari : kd\_instansi, kd\_subbidang, id\_tenagaahli, id\_pemberian jasa, kd\_waktu, kd\_daerah, kd\_kabupaten, kd\_provinsi, dan kd\_ibukota. *Measure* pada fakta konsultan perencanaan terdiri dari , total\_instansi, total\_nilaiinstansi, dan total\_tenagaahli.

1. ***Storing Pre-Calculation In The Fact Table***

*Storing Pre-Calculation In The Fact Table* atau yang disebut juga menyimpan pre-Calculation dalam tabel faktaterdapat data yang merupakan kalkulasi awal. Menyimpan pre-Calculation dalam tabel fakta. Hasil dari kalkulasi awal tersebut disimpan dalam tabel fakta. Pada tahapan ini seluruh tabel fakta yang telah dipilih akan dikaji ulang satu persatu, hal ini dapat mengetahui apakah adanya kemungkinan untuk dilakukannya pra perhitungan yang terdapat pada tabel fakta. Berikut dapat dilihat tabel dibawah ini :

**Tabel 4.5 *Pre-Calculation* Fakta Fasilitas**

|  |  |
| --- | --- |
| **Variabel** | **Multidimensi** |
| Nama Fasilitas | *Count*(fakta\_fasilitas.kd\_fasilitas) |
| Jenis Fasilitas | *Count*(fakta\_fasilitas.kd\_jnsfasilitas) |
| Merek Fasilitas | *Count*(fakta\_fasilitas.kd\_mrkfasilitas) |
| Keterangan Fasilitas | *Count*(fakta\_fasilitas.kd\_ketfasilitas) |
| Kondisi Fasilitas | *Count*fakta\_fasilitas.kd\_kondisifasilitas) |
| Waktu | *Count* (fakta\_fasilitas.waktu)  *Average*(fakta\_fasilitas.waktu) |
| Daerah | *Count* (fakta\_fasilitas.kd\_daerah) |

Pada tabel 4.5 merupakan kalkulasi awal yang tersimpan pada tabel fakta fasilitas yang terdiri dari nilai *COUNT* dan *AVERAGE* pada setiap *primary key* yang terdapat pada dimensi nama fasilitas, jenis fasilitas, merek fasilitas, keterangan fasilitas, kondisi fasilitas, waktu, dan daerah. Pada daerah memiliki data turunan atau yang disebut juga dengan *foreign key* yang terdiri dari kabupaten, provinsi dan ibukota. Dan beberapa measure yang akan didapat dari pengolahan dimensi pada tabel fakta fasilitas.

**Tabel 4.6 *Pre-Calculation* Fakta Tenaga Ahli**

|  |  |
| --- | --- |
| **Variabel** | **Multidimensi** |
| Nama tenaga ahli | *Count* (fakta\_tenagaahli.id\_tenagaahli) |
| Status tenaga ahli | *Count* (fakta\_tenagaahli.kd\_status) |
| Tingkat pendidikan | *Count* (fakta\_tenagaahli.kd\_tingkatpendidikan) |
| Daerah | *Count* (fakta\_tenagaahli.kd\_daerah) |
| Waktu | *Count* (fakta\_tenagaahli.kd\_waktu)  *Average* (fakta\_tenagaahli.kd\_waktu) |
| Jurusan | *Count* (fakta\_tenagaahli.kd\_jurusan) |
| Universitas | *Count* (fakta\_tenagaahli.kd\_universitas) |

Pada tabel 4.6 merupakan kalkulasi awal yang tersimpan pada tabel fakta tenaga ahli yang terdiri dari nilai *COUNT* dan *AVERAGE* pada setiap *primary key* yang terdapat pada dimensi nama tenaga ahli, status tenaga ahli, tingkat pendidikan, jurusan, univeristas, daerah, dan waktu.

**Tabel 4.7 *Pre-Calculation* Fakta Konsultan Perencaaan**

|  |  |
| --- | --- |
| **Variabel** | **Multidimensi** |
| Nama instansi | *Count* (fakta\_instansi.kd\_instansi |
| Tenaga ahli | *Count* (fakta\_instansi.tenagaahli) |
| Sub bidang tenaga ahli | *Count* (fakta\_instansi.kd\_subbidang) |
| Pemberi Jasa | *Count* (fakta\_instansi.id\_pemberijasa) |
| Waktu | *Count* (fakta\_instansi.kd\_waktu)  *Average* (fakta\_instansi.kd\_waktu) |
| Daerah | *Count* (fakta\_instansi.kd\_daerah) |

Pada tabel 4.7 merupakan kalkulasi awal yang tersimpan pada tabel fakta konsultan perencanaan yang terdiri dari nilai *COUNT* dan *AVERAGE* pada setiap *primary key* yang terdapat pada dimensi instansi, sub bidang, tenaga ahli, pemberi jasa, waktu, dan daerah.

1. ***Rounding Out The Dimension Tables***

*Rounding out the dimension tables* merupakan bagian untuk menambahkan deskripsi teks pada tabel dimensi. Agar deskripsi dapat dimengerti oleh user. Berikut ini merupakan deskripsi teks dari tebel dimensi yaitu :

**Tabel 4.8 *Rounding Out The Dimension Tables***

|  |  |  |
| --- | --- | --- |
| **Dimensi** | **Atribut** | **Deskripsi** |
| Daerah | Kd\_daerah  Nm\_daerah | Pada laporan dapat dilihat daerah dan lokasi proyek atau instansi, dan dapat mengetahui lokasi alamat tenaga ahli lahir, dan pemberian jasa |
| Waktu | Kd\_waktu  Waktu  Hari  Bulan  Tahun | Pada laporan dapat dilihat berdasarkan periode waktu yang dapat ditentukan, baik berdasarkan tanggal, hari, bulan dan tahun |
| Nama\_Fasilitas | Kd\_fasilitas  Nm\_fasilitas | Pada laporan dapat dilihat nama fasilitas yang ada pada perusahaan baik berupa elektronik yang digunakan ataupun kendaraan. |
| Jenis\_Fasilitas | Kd\_jnsfasilitas  Jns\_fasilitas | Pada laporan dapat dilihat jenis fasilitas digunakan pada perusahaan. |
| Merek\_Fasilitas | Kd\_mrkfasilitas  Merek\_fasilitas | Pada laporan dapat dilihat merek fasilitas yang sering digunakan oleh perusahaan dan melakukan perbandingan antara merek satu dengan yang lainnya. |
| Ket\_Fasilitas | Kd\_ketfasilitas  Ket\_fasilitas | Pada laporan dapat dilihat dibagian keterangan fasilitas apakah fasilitas merupakan sewa atau milik pribadi |
| Nama\_TenagaAhli | Id\_tenagaahli  Nama\_tenagaahli | Pada laporan dapat dilihat nama tenaga ahli yang digunakan oleh perusahaan dan melihat data lengkap tenaga ahli. |
| Tingkat\_Pendidikan | Kd\_tpendidikan  Jenis\_tpendidikan  Univeritas  Jurusan | Pada laporan dapat dilihat tingkat pendidikan tenaga ahli agar memberikan keputusan berdasarkan tingkatan pendidikannya. |
| Status\_TenagaAhli | Kd\_status  Status\_tenagaahli | Pada laporan dapat dilihat apakah tenaga ahli merupakan tenaga ahli tetap atau non-tetap |
| Jurusan | Kd\_jurusan  Nm\_jurusan | Pada laporan yang ada dapat dilihat jurusan tenaga ahli permasing-masing, agar dapat menentukan bidang pekerjaan yang layak untuk tenaga ahli. |
| Univeristas | Kd\_universitas  Nama\_universitas | Pada laporan dapat dilihat daftar asal lulusan universitas tenaga ahli. |
| Subbidang\_Instansi | Kd\_sbinstansi  Nm\_sbinstasi | Pada laporan dapat dilihat sub instansi yang diterima oleh tenaga ahli daam sebuah proyek. |
| Instansi | Kd\_instansi  Nm\_instansi | Pada laporan dapat dilihat nama instansi yang dikerjakan dalam perusahaan. |
| Pemberi\_jasa | Id\_pemberijasa  Nm\_pemberijasa  Alamat\_pemberijasa | Pada laporan dapat dilihat pemimpin atau pemberi jasa pada isntansi yang dikerjakan. |
| Provinsi | Kd\_provinsi  Nama\_provinsi | Pada laporan dapat dilihat informasi berupa provinsi yang sering diterima dalam proyek instansi. |
| Ibu Kota | Kd\_ibukota  Nama\_ibukota | Pada laporan dapat dilihat informasi ibu kota yang terdapat pada provinsi dalam pengerjaan proyek instansi. |
| Kabupaten | Kd\_kabupaten  Nama\_kabupaten | Pada laporan dapat dilihat informasi rincian daerah kabupaten yang dilakukannya pekerjaan proyek instansi perusahaan. |

Tabel 4.8 mendeskripsikan seluruh dimensi yang ada pada tabel fakta, dimensi tersebut terdiri dari dimensi waktu, daerah, provinsi, ibu kota, kabupaten, jenis fasilitas, merek fasilitas, keterangan fasilitas, tenaga ahli, tempat lahir tenaga ahli, tanggal lahir tenaga ahli, tingkat pendidikan, status tenaga ahli, jurusan tenaga ahli, universitas tenaga ahli, sub bidang instansi, nama instansi, dan pemberi jasa.

1. ***Choosing The Duration Of The Database***

Pada pemilihan durasi *database*, data yang akan dikelola kedalam *data warehouse* merupakan data yang bersifat *history*. Durasi yang akan digunakan tahun 2017 – 2013. Pada hal ini perlu diperhtikan tingkat akurasi yang dimiliki oleh data *history* dengan memperhatikan isi dan format data yang ada.

**Tabel 4.9 *Choosing The Duration Of The Database***

|  |  |  |  |
| --- | --- | --- | --- |
| **Nama Aplikasi** | **Database** | **Waktu Data yang Masuk Kedalam *Data warehouse*** | **Durasi *Data warehouse*** |
| CV.RGC | RGC | 2014-2018 | 5 TAHUN |

Pada tabel 4.9 menjelaskan durasi dari data yang akan diolah, durasi waktu yang digunakan memiliki rentang waktu 5 tahun yang terdiri dari 2014-2018.

1. ***Determine The Need To Track Slowly Changing Dimension***

*Detemine the need to track slownly changing dimension* atau dikenal mengawasi perubahan dimensi memiliki perubahan dimensi terhadap atribut dari tabel dimensi yang tidak selamanya memiliki nilai yang tetap atau bersifat relatif statis. Perubahan nilai atribut dapat terjadi dalam waktu yang cukup lama. dapat ditanggulangi yang terdiri dari 3 macam cara yaitu mangganti secara langsung pada tabel dimensi, pembentukkan *record* baru untuk perubahan baru yang terjadi disetiap waktu, dan perubahan data untuk membentuk kolom baru yang berbeda. Namun pada penelitian ini menggunakan cara kedua yaitu data baru akan dimasukkan kedalam *record* baru dan *record* lama akan tetap dan tidak dihapus. Atribut dari tabel dimensi tidak selamanya memiliki nilai tetap atau nilai yang bersifat relatif statis. Perubahan pada nilai atribut dapat terjadi dalam waktu yang relatif lama. Beberapa atribut yang dapat dilihat sebagai berikut :

**Tabel 4.10 Perbuahan Dimensi**

|  |  |  |
| --- | --- | --- |
| **No** | **Dimensi** | **Atribut yang Dapat Berubah** |
| 1 | Dim\_namafasilitas | Nama\_fasilitas |
| 2 | Dim\_merekfasilitas | Merek\_fasilitas |
| 3 | Dim\_jenisfasilitas | Jenis\_fasilitas |
| 4 | Dim\_tenagaahli | Nama\_tenagaahli |
| 5 | Dim\_status | Status\_tenagahli |
| 6 | Dim\_universitas | Nama\_Universitas |
| 7 | Dim\_instansi | Nama\_Instansi |
| 8 | Dim\_pemberijasa | Nama\_pemberijasa |
| 10 | Dim\_daerah | Daerah |
| 11 | Dim\_ketfasilitas | Keterangan\_fasilitas |
| 12 | Dim\_kondisifasilitas | Kondisi\_fasilitas |
| 13 | Dim\_subbidang | Sub\_bidang |

Pada tabel 4.10 merupakan dimensi yang yang dapat memasukan record baru tanpa mengubah dan menghapus record yang lama. Dimensi yang dapat melakukan perubahan yaitu dimensi fasilitas, merek fasilitas, jenis fasilitas, tenaga ahli, status, universitas, instansi, pemberi jasa, keterangan fasilitas, kondisi fasilitas, sub bidang dan daerah.

1. ***Dicide The Physical Design***

Dicide the physical design merupakan proses perancangan fisik *data warehouse*. Skema yang digunakan dalam *design* penelitian ini yaitu *snowflake schema* dan pada *data warehouse* menggunakan proses ETL ( *extract, Transform, Loading)* dan proses backup secara bekala.

1. **Proses ETL (*Extract Transform and Loading*)**
   1. Proses *Extract*

Proses *extract* merupakan proses dalam pengambilan data dari data sumber. Dalam penelitian ini dilakukan manual. Data sumber dapat beruba data lembaran *Excel* maupun data yang berformat file yang dihasilkan dari transaksi operasional kedalam satu bentuk database yang sama. Data tersebut dimasukan kedalam database *SQL Server.*

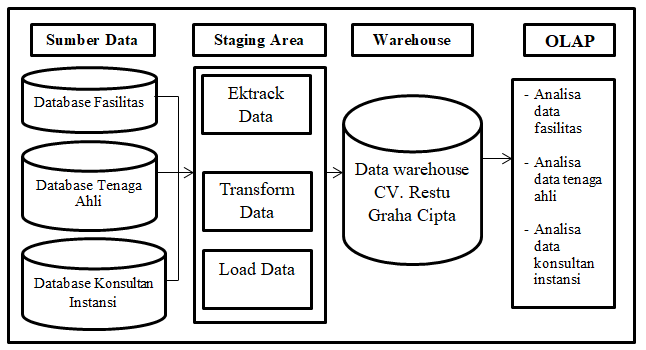
* 1. Proses *Transform*

Proses *Transform* yaitu proses lanjutan dari proses *ekstract*. Setelah melakukan peng-esktrackan maka data akan ditransform.

Hal-hal yang berkaitan dengan proses *transform* yaitu :

1. Memilih kolom atau *field* apa yang akan digunakan atau tidak.
2. Melakukan penerjamahan nilai-nilai yang berupa kode atau melakukan pengkodean sebuah nilai menjadi bentuk bebas. Contohnya dari laki-laki menjadi L dan Perempuan menjadi P.
3. Membuat sebuah ringkasan dari sekumpulan baris data.
   1. Proses *Loading*

Proses *loading* merupakan fase terakhir dari tahapan pengumpulan data dalam sebuah *data warehouse*. Fase yang memasukkan data kedalam target yaitu tabel-tabel ke *data warehouse*. Proses ETL dalam penelitian ini dilaksanakan setiap bulan. Karena data yang dilaksanakan merupakan data konsultan instansi yang proyek diterima dapat terjadi setiap bulannya. Dan laporan tersebut akan disusun pertahun. Proses ETL yang dilakukan dibawah pengawasan dan wewenang bagian pusat data dan informasi.



**Gambar 4.5 Rancangan Arsitektur *Data warehouse* CV. Restu Graha Cipta**

Pada gambar 4.5 diatas ini merupakan gambaran dari proses ETL (*extract, transform, and load*) pada perusahaan CV. RESTU GRAHA CIPTA. Sumber data yang akan diolah merupakan sumber data yang keseluruhan mencangkup pada data fasilitas, tenaga ahli dan konsultan perencanaan. Data tersebut merupakan data mentah dari perusahaan yang belum diolah dan memiliki berbagai macam format. Data tersebut akan dimasukkan kedalam *database SQL Server*. Kemudian data tersebut di *ekstract* yaitu proses penguraian dan pembersihan data untuk mendapatkan strukur data yang diinginkan. Dimana data diseleksi atau dilakukan penyaringan terhadap data hasil ekstraksi tersebut. *Transfrom* merupakan langkah kedua dalam proses. Pada proses ini mengubah data dari format operasional menjadi format *data warehouse* seperti mengkonversi tipe data, pembersihan serta pembuangan duplikasi dan kesalahan data, melakukan perhitungan nilai-nilai derivat atau mula-mula dan pemeriksaan integritas referensi data. *Loading* merupakan langkah terakhir pada proses ETL yaitu proses untuk memasukkan data kedalam target terakhir yaitu *data warehouse*. Setelah proses ETL selesai maka data akan dimasukkan kedalam *data warehouse* CV. Restu Graha Cipta. Hasil dari *data warehouse* dimasukkan kedalam OLAP untuk menghasilkan informasi multidimensi yang data tersebut dapat dilihat dari berbagai sudut pandang.

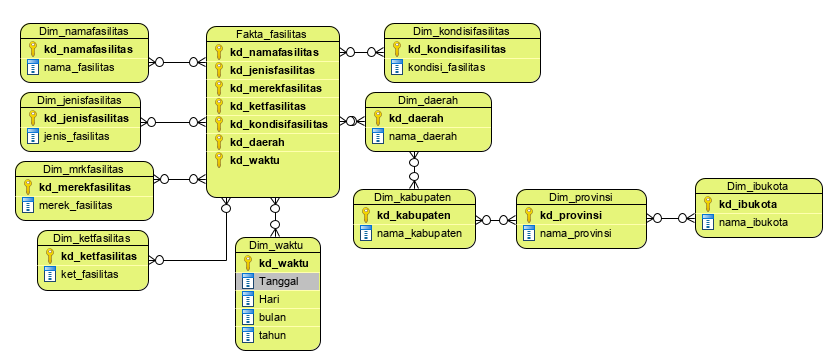
# Perancangan

# Perancangan Skema *Data Warehouse*

Skema yang digunakan pada *data warehouse* perusahaan CV. Restu Graha Cipta yaitu *schema snowflake. Schema snowflake* pada *data warehouse* ini bertujuan untuk menormalisasikan tabel-tabel dimensi yang berukuran besar dengan satu atau lebih kolom yang memiliki duplikasi terhadap data. Dimensi yang telah dinormalisasikan untuk mengurangi redudansi data sehingga struktur data tabelnya mudah dibaca dan dimengerti. Dibawah ini merupakan gambaran *schema snowflake* terhadap dimensi yang ada pada *data waerehose* perusahaan CV. Restu Graha Cipta.

1. *Schema Snowflake* Tabel Fakta Fasilitas

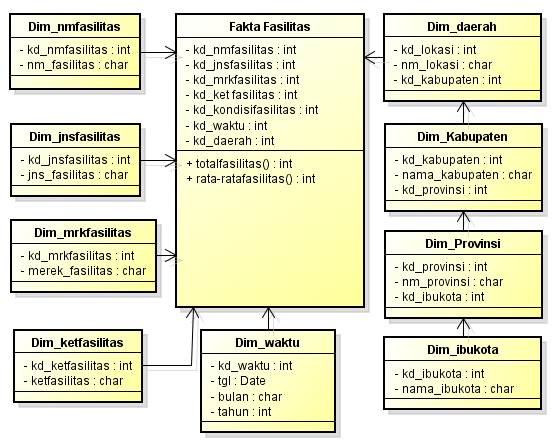
Pada *schema snowflake* tabel fakta fasilitas terdapat *entity relationship diagram* fakta fasilitas yang berguna untuk mendeskripsikan data-data pada penyimpanan yang saling terhubung satu sama lain. Berikut ini merupakan gambar *entity relationship diagram* pada fakta fasilitas dibawah ini :



**Gambar 4.6 Entity Relationship Diagram Fakta Fasilitas**

Pada Gambar 4.6 entitas diatas menjelaskan hubungan data-data yang terdapat pada tabel fakta fasilitas. Data-data yang terdapat pada fakta fasilitas yaitu terdiri dari data nama fasilitas, jenis fasilitas, merek fasilitas, keterangan fasilitas, kondisi fasilitas, waktu, daerah memiliki *foreign key* yang merupakan data turunan dari kabupaten. provinsi dan ibukota.

*Schema snowflake* pada tabel fakta fasilitas menjelaskan tentang dimensi yang terhubung pada tabel fakta, dimensi tersebut dinormalisasikan untuk mengurangi redudansi terhdapat fakta yang ada. Dapat dilihat pada gambar dibawah ini :

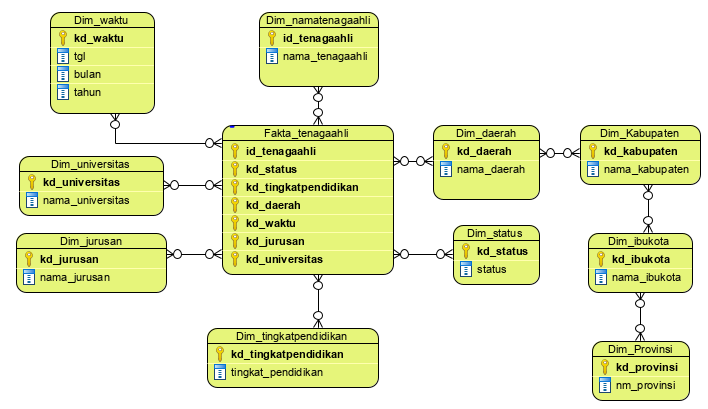


**Gambar 4.7 Skema Snowflakes Fasilitas**

Pada gambar 4.7 *schema snowflakes* pada fakta tabel fasilitas, yang terdiri dari beberapa dimensi yang saling terhubung ketabel fakta dan dimensi yang terhubung dengan dimensi lainnya. Dimensi tersebut antara lain : nama fasilitas, jenis fasilitas, merek fasilitas, keterangan fasilitas, kondisi fasilitas, waktu, daerah yang didalamnya ternormalisasikan dimensi provinsi, dimensi ibu kota, dan dimensi kabupaten. Pada skema fasilitas terdapat *measure* yang terdiri dari total fasilitas dan jumlah fasilitas. Pada dimensi daerah memiliki *foreign key* dari dimensi kabupaten, sedangkan kabupaten memiliki *foreign key* dari provinsi dan provinsi memiliki *foreign key* dari ibukota.

1. *Schema snowflake* Tabel Fakta Tenaga Ahli

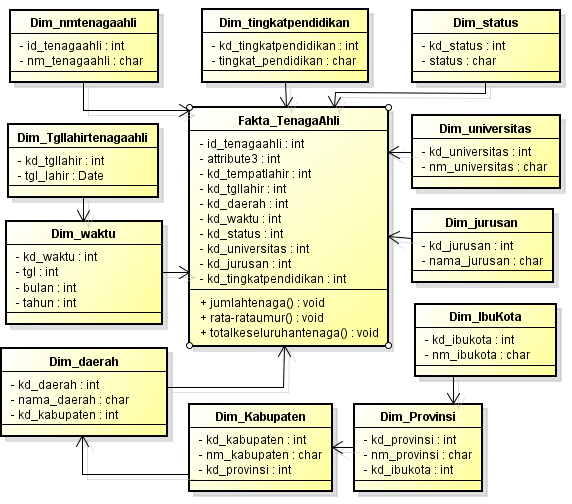
Pada *schema snowflake* tabel fakta tenaga ahli terdapat *entity relationship diagram* fakta tenaga ahli yang berguna untuk mendeskripsikan data-data pada penyimpanan yang saling terhubung satu sama lain. Berikut ini merupakan gambar *entity relationship diagram* pada fakta tenaga ahli dibawah ini :



**Gambar 4.8 Entitas Relationship Diagram Tenaga Ahli**

Pada Gambar 4.8 entitas diatas menjelaskan hubungan data-data yang terdapat pada tabel fakta tenaga ahli. Data-data yang terdapat pada fakta tenaga ahli terdiri dari data nama tenaga ahli, status, tingkat pendidikan, universitas, jurusan, daerah, provinsi, ibukota, kabupaten dan waktu.

*Schema snowflake* pada tabel fakta tenaga ahli menjelaskan tentang dimensi yang terhubung pada tabel fakta, dimensi tersebut dinormalisasikan untuk mengurangi redudansi terhadapat fakta seperti gambar dibawah ini :

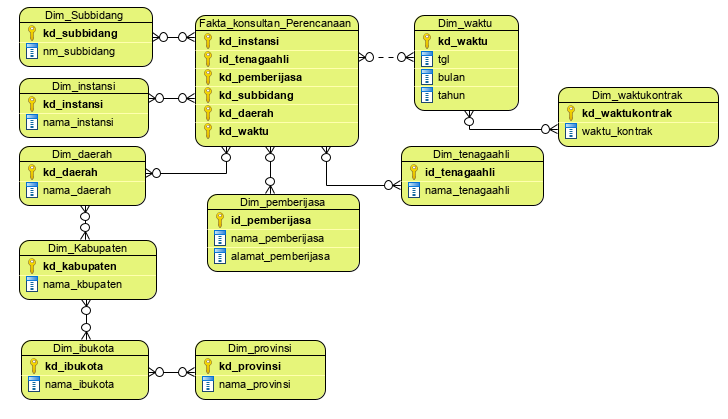


**Gambar 4.9 Skema *Snowflakes* Tenaga Ahli**

Pada gambar 4.9 *schema snowflakes* tabel fakta tenaga ahli pada perusahaan. Pada tabel fakta terdiri dari eberapa dimensi yang terhubung dengan tabel fakta dan dimensi yang saling terhubung satu sama lain. Dimensi yang terdapat pada tabel fakta tenaga ahli terdiri dari : nama tenaga ahli, tenaga ahli, daerah yang terdiri dari dimensi provinsi, ibu kota dan kabupaten, universitas yang terdiri dari dimensi jurusan dan tingkat pendidikan, dimensi status dan dimensi waktu. Pada dimensi daerah memiliki *foreign key* dari dimensi kabupaten, sedangkan kabupaten memiliki *foreign key* dari provinsi dan provinsi memiliki *foreign key* dari ibukota.

1. *Schema snowflake* Tabel Fakta Konsultan Instansi

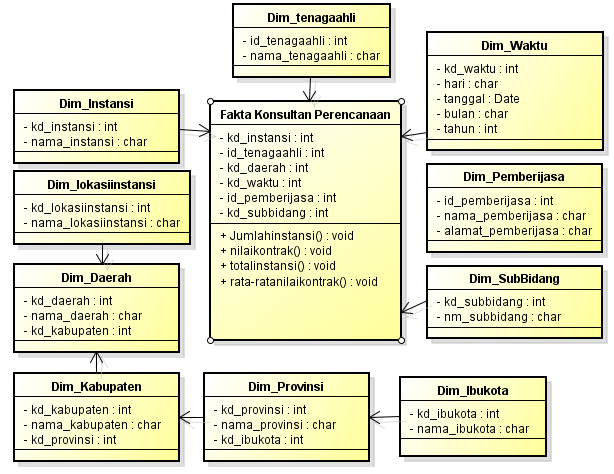
Pada *schema snowflake* tabel fakta konsultan instansi terdapat *entity relationship diagram* fakta konsultan instansi yang berguna untuk mendeskripsikan data-data pada penyimpanan yang saling terhubung satu sama lain. Berikut ini merupakan gambar *entity relationship diagram* pada fakta konsultan instansi dibawah ini :



**Gambar 4.10 Entity Relationship Diagram Konsultan Perencanaan**

Pada Gambar 4.10 entitas diatas menjelaskan hubungan data-data yang terdapat pada tabel fakta konsultan perencanaan. Data-data yang terdapat pada fakta konsultan perencanaan yaitu terdiri dari data nama instansi, tenaga ahli, sub bidang, pemberi jasa, daerah, provinsi, ibukota, kabupaten, waktu.

*Schema snowflake* pada tabel fakta konsultasi pembangunanmenjelaskan tentang dimensi yang terhubung pada tabel fakta, dimensi tersebut dinormalisasikan untuk mengurangi redudansi terhdapat fakta yang ada. Dapat dilihat pada gambar dibawah ini :

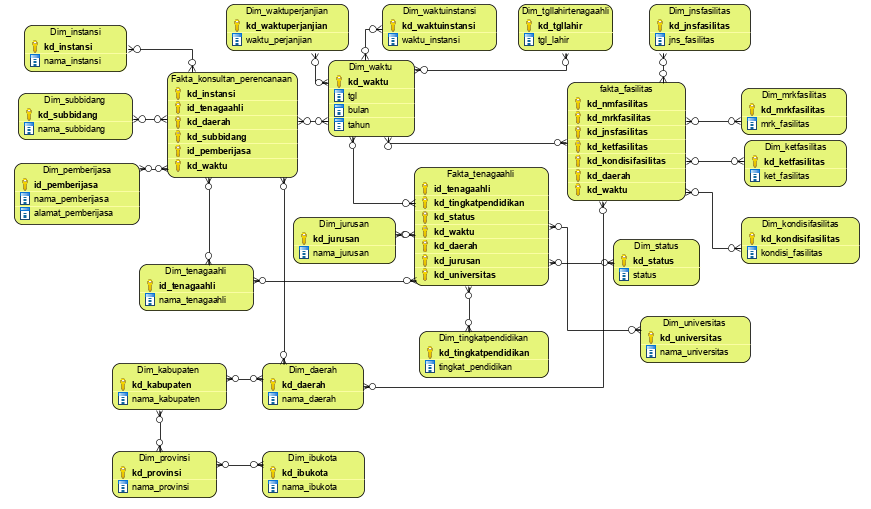


**Gambar 4.11 Skema Snowflakes Konsultan Perencanaan**

Pada gambar 4.11 *schema snowflakes* tabel fakta konsultan perencanaan pada perusahaan. Tabel fakta tersebut terdiri dari beberapa dimensi yang terhubung ke tabel fakta serta beberapa dimensi yang saling terhubung satu sama lain. Dimensi yang terdapat pada tabel fakta konsultan perencanaan terdiri dari : dimensi instansi, dimensi tenaga ahli, dimensi sub bidang, dimensi waktu, daerah, provinsi, ibu kota dan kabupaten, dan pemberian jasa yang terdiri dari alamat pemberi jasa. Pada dimensi daerah memiliki *foreign key* dari dimensi kabupaten, sedangkan kabupaten memiliki *foreign key* dari provinsi dan provinsi memiliki *foreign key* dari ibukota.

1. *Schema snowflake* *Data warehouse* CV. Restu Graha Cipta

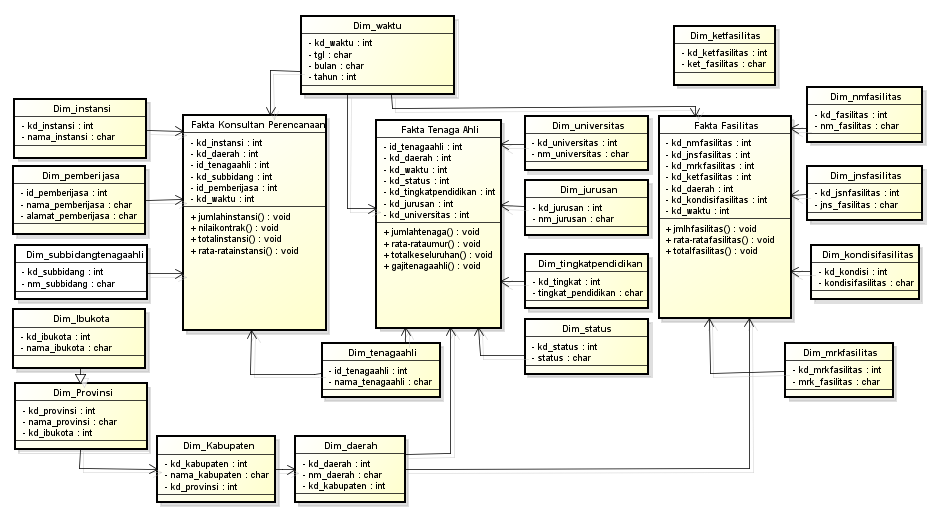
Pada *schema snowflake* *data warehouse* CV. Restu Graha Cipta terdapat *entity relationship diagram* *data warehouse* tersebut yang berguna untuk mendeskripsikan data-data pada penyimpanan yang saling terhubung satu sama lain. Berikut ini merupakan gambar *entity relationship diagram* pada *data warehouse* CV. Restu Graha Cipta dibawah ini :



**Gambar 4.12 ERD *Data warehouse* CV Restu Graha Cipta**

Pada Gambar 4.12 entitas diatas menjelaskan hubungan data-data yang terdapat pada *data warehouse* CV. Restu Graha Cipta. Data-data yang terdapat pada *data warehouse* CV. Restu Graha Cipta yaitu terdiri dari seluruh data yang ada pada fakta fasilitas, fakta tenaga ahli dan konsultan perencanaan.

*Schema snowflake* pada *data warehouse* CV. Restu Graha Cipta menjelaskan tentang keseluruhan dimensi yang terhubung pada seluruh tabel fakta, dimensi tersebut dinormalisasikan untuk mengurangi redudansi terhdapat fakta yang ada. Dapat dilihat pada gambar dibawah ini :



**Gambar 4.13 Schema *Snowflake* *Data warehouse* CV Restu Graha Cipta**

Pada gambar 4.13 *Schema snowflakes* *dari data warehouse* secara keseluruhan di perusahaan CV. Restu Graha Cipta yang terdiri dari tabel fakta fasilitas, tenaga ahli dan konsultan perencanaan. Dimensi yang saling berhubungan satu sama lain dan dimensi yang terhubung dengan tabel fakta, yang terdiri dari dimensi : nama fasilitas, jenis fasilitas, merek fasilitas, keterangan fasilitas, kondisi fasilitas, waktu, lokasi fasilitasi, nama tenaga ahli, tgl lahir tenaga ahli, tempat lahir tenaga ahli, daerah yang terdiri dari dimensi provinsi, ibu kota dan kabupaten, universitas yang terdiri dari dimensi jurusan, tingkat pendidikan dan daerah, dimensi status, dimensi instansi, dimensi sub bidang, dimensi waktu yang terdiri dari waktu instansi dan waktu perjanjian, lokasi instansi, dan pemberian jasa yang terdiri dari no telp pemberi jasa.

# Perancangan UML (*Unified Modelling Language*)

Aplikasi yang akan peneliti rancang dimodelkan dengan menggunakan alat bantu, yaitu UML (*Unified Modelling Language*) untuk menentukan cara kerja program “*Perancangan Data Warehouse dengan Menggunakan Metode Extract, Transform, dan Load di CV. Restu Graha Cipta Padang*” yang menggunakan Use Case Diagram, Class Diagram, Sequence Diagram, State Diagram, Collaboration Diagram, Activity Diagram dan Deployment Diagram. Perancangan menggunakan program Astah Professional. Adapun perancangan UML (*Unified Modelling Language*) untuk perancangan data warehouse sebagai berikut :

* + - 1. **Use Case Diagram**

Use Case Diagram menggambarkan bagaimana seseorang akan menggunakan atau memanfaatkan sistem, sedangkan aktor adalah seseorang atau seseuatu yang berinteraksi dengan sistem. Use Case Diagram menggambarkan bagaimana proses-proses yang dilakukan oleh aktor terhadap sebuah sistem.

* + - * 1. Definisi Actor

Definisi aktor adalah seluruh aktivitas yang dilakukan dalam menggunakan sistem aplikasi, definisi aktor dapat dilihat pada tabel 4.11 berikut ini :

**Tabel 4.11 Definisi Aktor**

|  |  |
| --- | --- |
| **Aktor** | **Deskripsi** |
| Admin | Admin dapat melakukan seluruh kegiatan yang ada dalam sistem |
| Owner | Pemimpin dapat melihat data yang terdapat dalam sistem. |

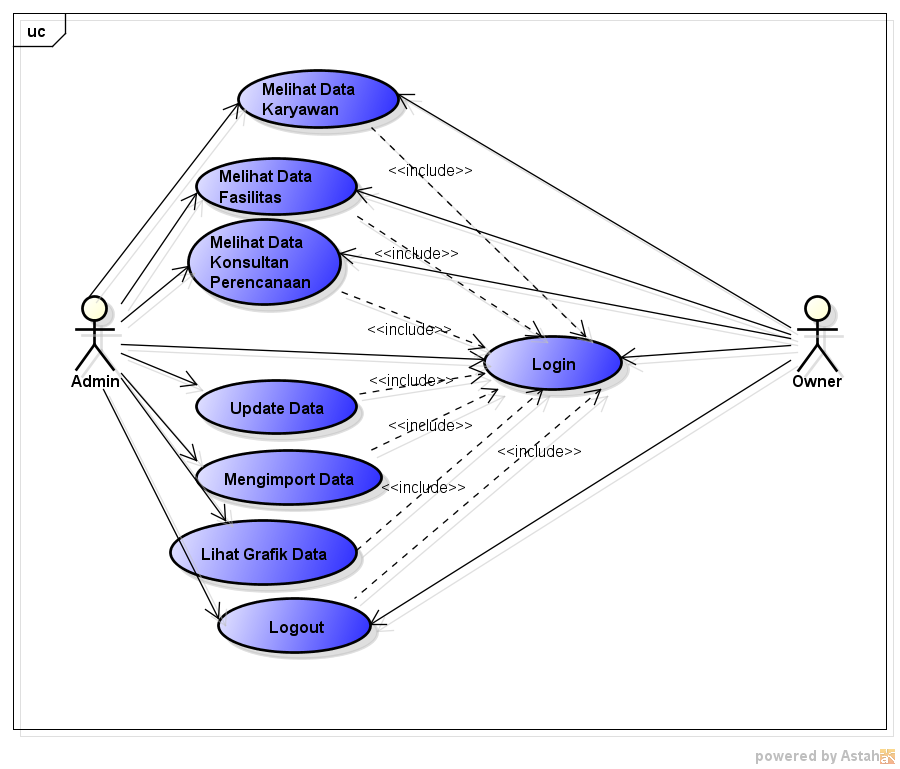
* 1. Definisi Use Case

Definisi use case merupakan kegiatan-kegiatan yang dapat dilakukan oleh aktor dalam mengakses sistem. Pendefinisian use case dapat dilihat pada tabel 4.12 dibawah ini :

**Tabel 4.12 Definisi Use Case**

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Use Case** | **Deskripsi** | **Actor** |
| 1 | Login | Proses agar actor dapat masuk dan melakukan aktifitas kedalam sistem | Admin |
| 2 | Lihat Data | Admin dan owner dapat melihat data. | Admin, Owner |
| 4 | Update Data | Proses dalam melakukan pengupdatean data yang dikelola berdasarkan waktu yang diinginkan | Admin |
| 5 | Import Data | Sebelum melakukan penganalisaan data maka, terlebih dahulu melakukan import data kedalam sistem. | Admin |
| 6 | Lihat grafik pengembangan data | Melakukan penganalisaan data dengan melihat pengembangan grafik data | Admin |
| 7 | Logout | Proses untuk meninggalkan aktivitas yang ada didalam sistem. | Admin |

Tabel 4.12 diatas merupakan kegiatan-kegiatan yang dapat dilakukan oleh admin dalam mengakses sistem. Owner juga memiliki kegiatan untuk melihat informasi data yang ada pada sistem. Agar lebih jelas dalam sistem dapat dilihat pada gambar 4.8 Use Case Diagram berikut :



**Gambar 4.14 Use Case Diagram**

Pada gambar 4.14 diatas yaitu *use case diagram* yang terdiri dari peran dan aktifitas apa saja yang didapatkan dilakukan oleh admin dan owner sebagai aktor dalam sistem, pada gambar diatas terlihat atribut paling utama dari sistem adalah *login*. Maka apabila telah melakukan *login*, hal ini membuat admin dapat melakukan aktifitas-aktifitas serta proses didalam sistem. Aktifitas-aktifitas yang dilakukan oleh admin antara lain melakukan melihat data, melakukan update data, melakukan import data, melakukan penganalisaan pengembangan dengan melihat grafik data dan melakukan logout untuk keluar dalam sistem. Owner dapat melakukan aktifitas untuk melihat data yang ada pada sistem.

# *Class Diagram*

*Class diagram* menggambarkan keadaan (atribut/property) suatu sistem, sekaligus menambahkan kelas tabel, *field* dan operasi-operasi lainnya dalam program databaase yang digunakan nantinya. Class diagram merupakan penjelasan proses database dalam suatu program. Class diagram membantu dalam memvisualisasikan struktur kelas dari suatu sistem dan merupakan tipe diagram yang paling ditemui dalam pemodelan system berbasis object-oriented.

* 1. Definsi *Class*

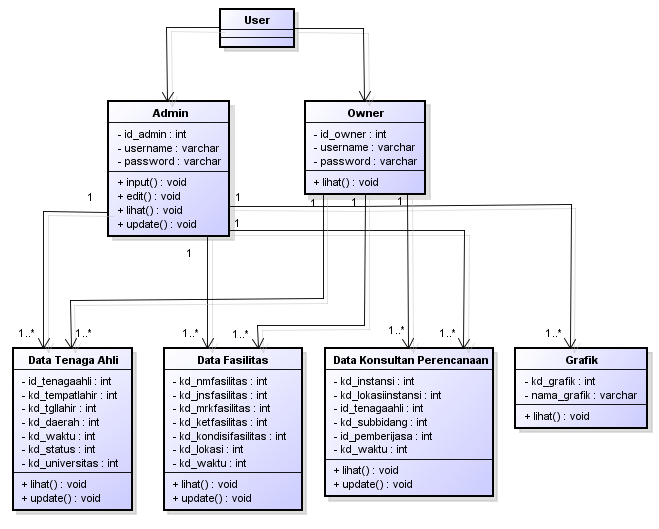
Class merupakan spesifikasi yang menghasilkan sebuah objek dan merupakan dasar dari pengembangan dan desain program yang berorientasi objek. Class menggambarkan keadaan (atribut/properti) dari suatu sistem, sekaligus layanan untuk memanipulasi keadaan tersebut (metoda/fungsi). *Class diagram* menggambarkan struktur dan deskripsi class, package dan objek beserta hubungan satu sama lain seperti *containment*, pewarisan, asosiasi, dan lain-lain. Hal ini dapat dilihat pada tabel 4.13 berikut ini :

**Tabel 4.13 *Class diagram***

|  |  |  |
| --- | --- | --- |
| **No** | **Class** | **Deskripsi** |
| 1 | User | Melakukan proses keseluruhan data yang dari sistem pada perusahaan CV. Restu Graha Cipta. |
| 2 | Data | Untuk menyimpan seluruh data yang akan diolah yang terdiri dari :   1. Data Fasilitas 2. Data Tenaga Ahli 3. Data Konsultan Perencanaan |
| 3 | Grafik | Untuk melakukan proses penganalisaan dari hasil data yang telah diolah |

Dari tabel 4.13 diatas dapat menghasilkan sebuah objek dan merupakan dasar dari pengembangan dan desain berorientasi objek. *Class* yang ada pada objek terdiri dari *user*, data dan grafik. Data yang ada pada objek merupakan data fasilitas, tenaga ahli dan konsultan perencanaan.

*Class diagram* yang digunakan dalam sistem perusahaan CV. Restu Graha Cipta untuk melihat atribut yang ada pada perusahaan dapat dilihat pada gambar 4.15 dibawah berikut ini :



**Gambar 4.15 *Class diagram***

Pada gambar 4.15 diatas mendefiniskan hubungan antar kelas-kelas yang ada dalam *database* yang terdiri dari : *admin, owner*, data tenaga ahli, data konsultan perencanaan, data fasilitas, grafik yang tabel tersebut saling terhubung satu sama lain, yang hasil data tersebut akan membentuk grafik analisa data.

1. Rancangan *Database*

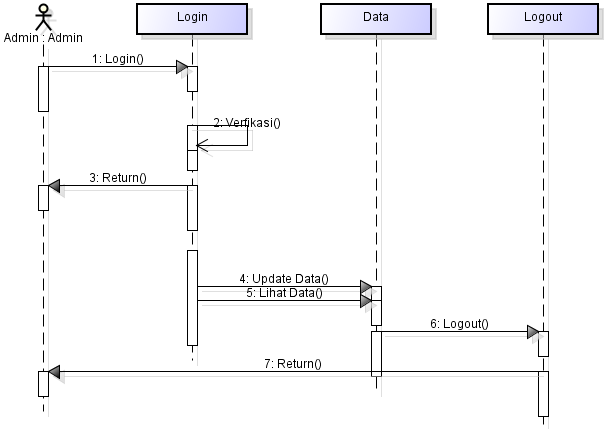
Perancangan *database* yang dilakukan setelah permodelan dibuat dengan meggunakan *Mysql* sebagai database sebagai tempat penyimpanan *data warehouse*.

# *Sequence Diagram*

*Sequence diagram* menggambarkan interaksi antar objek didalam dan diluar sekitar sistem baik aktor maupun display dengan berupa pesan yang digambarkan terhadap waktu. *Sequence diagram* digunakan untuk menggambarkan prilaku pada skenario secara detail menurut waktu. Komponen utama *sequence* diagram terdiri dari objek yang ditulis dalam kotak segiempat atau *participant.* Pembuatan *sequence diagram* ini merupakan aktifitas yang kritikal dari proses perancangan yang nantinya akan menjadi pedoman dalam proses pemograman yang berisi aliran kontrol dari program.

* 1. *Sequence Diagram* Untuk Pengolahan Data Admin

*Sequence diagram* pengolahan data menjelaskan tentang seluruh kegiatan pengolahan data yang terdapat pada sistem yang dilakukan oleh admin.

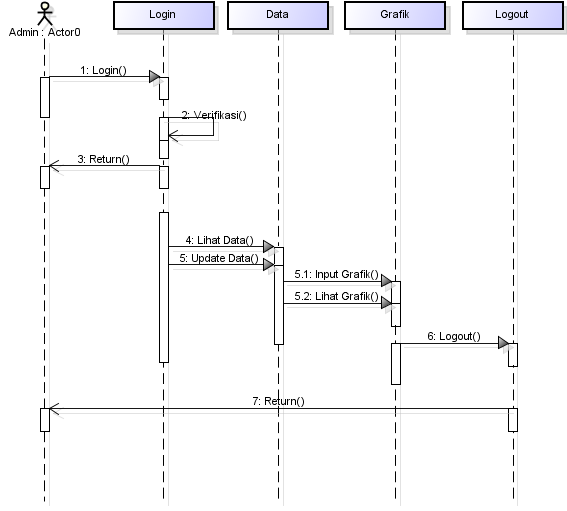


**Gambar 4.16 Sequence Diagram Pengolahan Data Admin**

Pada gambar 4.16 menggambarkan interaksi aktor pada sistem yang akan dilakukan oleh aktor admin dalam melakukan pengolahan data yang terdapat pada sistem dengan mengisi form *login* terlebih dahulu, jika salah maka akan sistem akan melaukan penolakan dan admin mengulangi proses pengisisan form *login*. Jika benar maka admin akan masuk kedalam menu *home* kemudian admin dapat melakukan pengolahan data yang ingin dilakukan.

* 1. *Sequence Diagram* Untuk Pengolahan Grafik Data

*Sequence diagram* untuk pengolahan grafik data menjelaskan pengembangan analisa data yang ada pada *data warehouse* untuk dijadikan sebagai sarana informasi yang berguna bagi perusahaan.

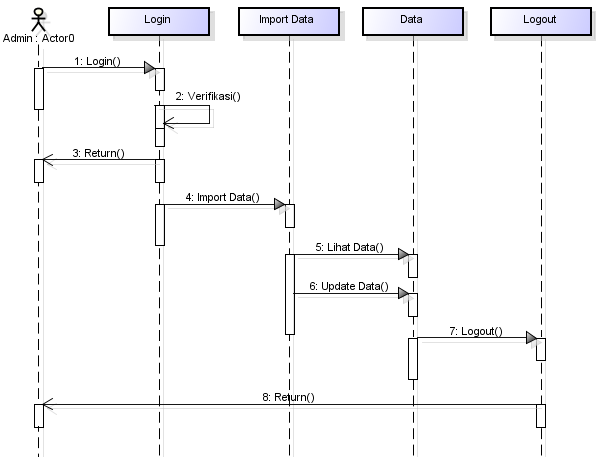


**Gambar 4.17 Sequence Diagram Grafik**

Pada gambar 4.17 menggambarkan interaksi aktor pada sistem yang akan dilakukan oleh aktor admin dalam melakukan pengolahan grafik data yang terdapat pada sistem dengan mengisi form login terlebih dahulu, jika salah maka akan sistem akan melakukan penolakan dan admin mengulangi proses pengisisan form login. Jika benar maka admin akan masuk kedalam menu home kemudian admin memilihan data untuk melakukan pengolahan grafik data yang ingin dilakukan.

1. Sequence Diagram Untuk Import Data

Sequence diagram untuk import data yaitu menjelaskan tentang kegiatan yang dilakukan oleh admin dalam melakukan import data kedalam sistem aplikasi yang digunakan. Dalam melakukan import data hanya boleh dilakukan oleh admin dari perusahaan, admin bertugas dalam pengolan data yang ada pada sistem. Sequence diagram untuk mengimport data dapat dilihat dibawah ini sebagai berikut:

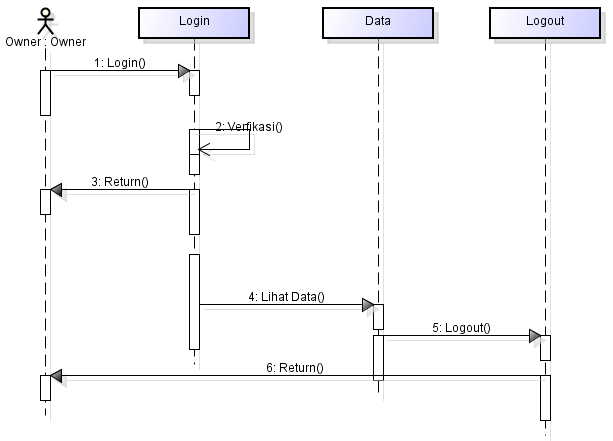


**Gambar 4.18 Sequence Diagram Import Data**

Pada gambar 4.18 admin dalam melakukan pengolahan data yang terdapat pada sistem dengan mengisi form login terlebih dahulu, jika salah maka akan sistem akan melaukan penolakan dan admin mengulangi proses pengisisan form login. Jika benar maka admin akan masuk kedalam menu home kemudian admin dapat melakukan pengolahan dalam mengimport data.

1. *Sequence Diagram* Pengolahan Data Pada Owner

*Sequence diagram* pengolahan datapada owner yaitu menjelaskan tentang kegiatan yang dilakukan oleh owner pada sistem.



**Gambar 4.19 Sequence Diagram Pengolahan Data Pada Owner**

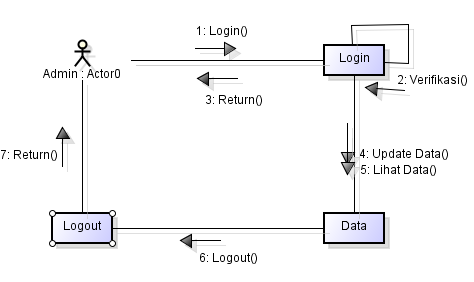
Pada gambar 4.19 menggambarkan interaksi aktor pada sistem yang akan dilakukan oleh aktor owner dalam melakukan pengolahan data yang terdapat pada sistem dengan mengisi form login terlebih dahulu, jika salah maka akan sistem akan melaukan penolakan dan admin mengulangi proses pengisisan form login. Jika benar maka owner akan masuk kedalam menu home kemudian admin dapat melakukan pengolahan data yang ingin dilakukan.

# *Collaboration Diagram*

*Collaboration diagram* menggambarkan *message-message* yang dikirim satu sama lainnya, serta untuk menunjukkan *physical view* dari suatu sistem yang akan dibangun.

* 1. *Collaboration Diagram* Pengolahan Data Admin

*Collaboration Diagram* pengolahan data mengambarkan serta menjelaskan urutan dalam melakukan proses pengolahan data yang terdapat dalam sistem. Berikut merupakan gambar collaboration diagram pengolahan data dibawah ini :

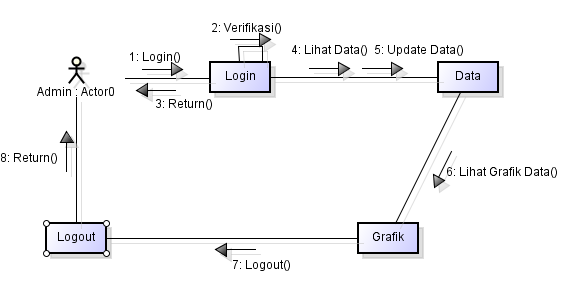


**Gambar 4.20 Collaboration Diagram Pengolahan Data Admin**

Pada gambar 4.20 admin dalam melakukan pengolahan data yang terdapat pada sistem dengan mengisi form login terlebih dahulu, jika salah sistem akan melakukan penolakan, admin kembali mengisi form login. Jika benar admin akan kemudian admin dapat melakukan pengolahan data.

* 1. *Collaboration Diagram* Pengolahan Grafik

*Collaboration diagram* pengolahan grafik yaitu menjelaskan proses dalam melakukan penganalisaan data yang diinputkan dari data terdapat dalam sistem. Grafik yang terdapat dalam sistem akan digunakan untuk menganalisa data yang akan diambil sebuah keputusan dalam perusahaan tersebut.

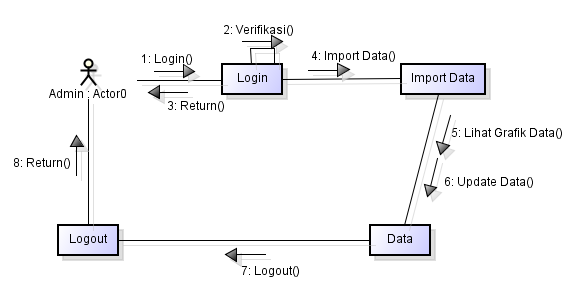


**Gambar 4.21 Collaboration Diagram Pengolahan Grafik Data**

Pada gambar 4.21 menggambarkan interaksi aktor pada sistem yang akan dilakukan oleh aktor admin dalam melakukan pengolahan grafik data yang terdapat pada sistem dengan mengisi form login terlebih dahulu, jika salah maka akan sistem akan melaukan penolakan dan admin mengulangi proses pengisisan form login. Jika benar maka admin akan masuk kedalam menu home kemudian admin memilihan data untuk melakukan pengolahan grafik data yang ingin dilakukan.

* 1. *Collaboration Diagram* Import Data

*Collaboration diagram* ini menjelaskan urutan dalam melakukan proses pengimporttan data yang ada didalam sistem. Yang dijelaskan dalam gambar 4.21 dibawah ini sebagai berikut :

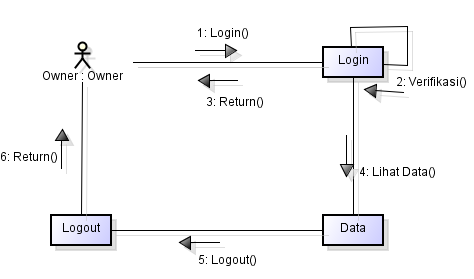


**Gambar 4.22 Collaboratin Diagram Import Data**

Pada gambar 4.22 admin dalam melakukan pengolahan data yang terdapat pada sistem dengan mengisi form login terlebih dahulu, jika salah maka akan sistem akan melaukan penolakan dan admin mengulangi proses pengisisan form login. Jika benar maka admin akan masuk kedalam menu home kemudian admin dapat melakukan pengolahan dalam mengimport data.

* 1. *Collaboration Diagram* Pengolahan Data Pada Owner

*Collaboration Diagram* pengolahan data pada owner yaitu menjelaskan tentang kegiatan yang dilakukan oleh owner pada sistem. Dapat dilihat pada gambar dibawah ini :



**Gambar 4.23 *Collaboration Diagram* Pengolahan Data Pada Owner**

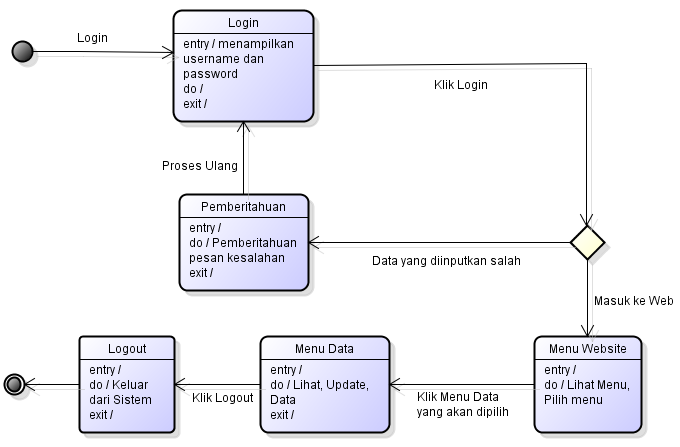
Pada gambar 4.23 owner melakukan pengisian form login terlebih dahulu, jika salah maka akan sistem akan melakukan penolakan dan owner mengulangi proses pengisisan form login. Jika benar maka owner akan masuk kedalam menu home kemudian admin dapat melihat data yang ingin dilakukan.

# Statechart atau State Machine Diagram

Statechart diagram atau machine diagram menggambarkan transisi dan perubahan keadaan yaitu antar satu state ke state lain pada suatu object dalam sistem.

* 1. Statechart Diagram Untuk Pengolahan Data Admin

Diagram ini menjelaskan tentang alur dalam proses pengolahan data yang dilakukan admin kedalam sistem. Untuk lebih jelasnya dapat dilihat pada gambar 4.24 berikut :

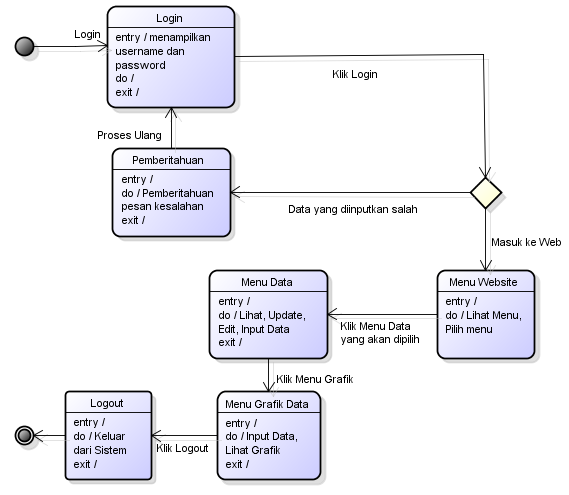


**Gambar 4.24 Statechart Diagram Pengolahan Data Admin**

Pada gambar 4.24 menggambarkan interaksi aktor pada sistem yang akan dilakukan oleh aktor admin dalam melakukan pengolahan data yang terdapat pada sistem dengan mengisi form login terlebih dahulu, jika salah maka akan sistem akan melaukan penolakan dan admin mengulangi proses pengisisan form login. Jika benar maka admin akan masuk kedalam menu home kemudian admin dapat melakukan pengolahan data yang ingin dilakukan.

* 1. Statechart Diagram Untuk Pengolahan Grafik Data

Statechart Diagram ini menjelaskan bagaimana alur tentang pengolahan grafik yang terjadi pada sistem *data warehouse* ini, pengolahan proses tersebut dilakukan oleh admin didalam sistem. Untuk lebih jelas dapat diligat pada gamabr 4.24 dibawah ini sebagai berikut :

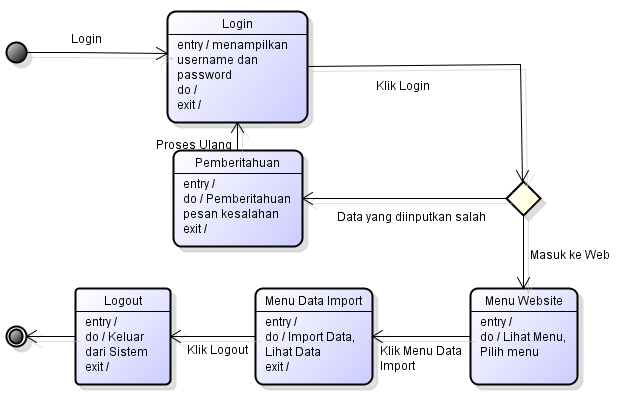


**Gambar 4.25 Statechart Diagram Pengolahan Grafik Data**

Pada gambar 4.25 menggambarkan interaksi aktor pada sistem yang akan dilakukan oleh aktor admin dalam melakukan pengolahan grafik data yang terdapat pada sistem dengan mengisi form login terlebih dahulu, jika salah maka akan sistem akan melakukan penolakan dan admin mengulangi proses pengisisan form login. Jika benar maka admin akan masuk kedalam menu home kemudian admin memilihan data untuk melakukan pengolahan grafik data yang ingin dilakukan.

* 1. Statechart Diagram Import Data

Statechart Diagram dalam import data menjelaskan tentang alur dalam proses pengimportan data yang dilakukan oleh admin. Untuk lebih jelasnya bisa dilihat pada gambar 4.26 berikut dibawah ini :

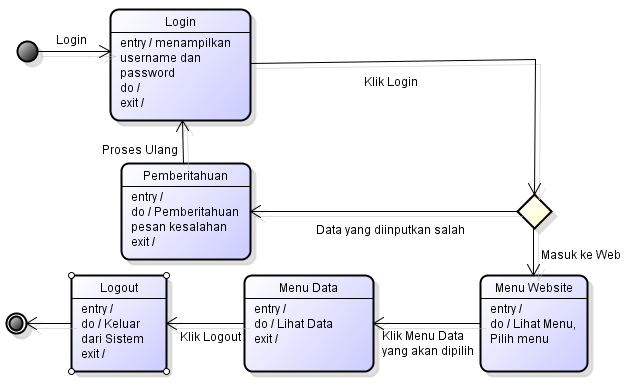


**Gambar 4.26 Statechart Diagram Import Data**

Pada gambar 4.26 menggambarkan interaksi aktor pada sistem yang akan dilakukan oleh aktor admin dalam melakukan pengolahan data yang terdapat pada sistem dengan mengisi form login terlebih dahulu, jika salah maka akan sistem akan melaukan penolakan dan admin mengulangi proses pengisisan form login. Jika benar maka admin akan masuk kedalam menu home kemudian admin dapat melakukan pengolahan dalam mengimport data.

1. Statechart Diagram Pengolahan Data Pada Owner

*Statechart Diagram* pengolahan data pada owner yaitu menjelaskan tentang kegiatan yang dilakukan oleh owner pada sistem. Dapat dilihat pada gambar dibawah ini :



**Gambar 4.27 Statechart Diagram Pengolahan Data Pada Owner**

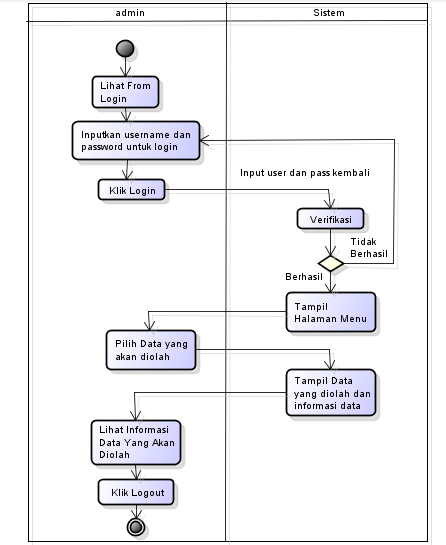
Pada gambar 4.27 menggambarkan interaksi aktor pada sistem yang akan dilakukan oleh aktor owner dalam melakukan pengolahan data yang terdapat pada sistem dengan mengisi form login terlebih dahulu, jika salah maka akan sistem akan melaukan penolakan dan owner mengulangi proses pengisisan form login. Jika benar maka owner akan masuk kedalam menu home kemudian admin dapat melakukan pengolahan data yang ingin dilakukan.

# Activity Diagram

Activity diagram menggambarkan tentang bagaimana aktivitas yang terjadi dalam sistem yang akan dirancang. Activity diagram hampir sama dengan flowchart diagram yang menggambarkan proses terjadinya antara aktor dan sistem. :

* 1. Activity Diagram Pengolahan Data Admin

Diagram ini menjelaskan tentang alur dalam proses pengolahan data yang dilakukan admin kedalam sistem.

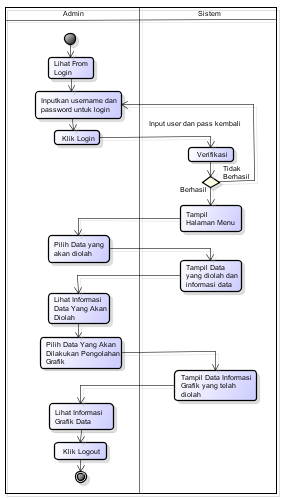


**Gambar 4.28 Activity Diagram Pengolahan Data Admin**

Pada gambar 4.28 menggambarkan interaksi aktor pada sistem yang akan dilakukan oleh aktor admin dalam melakukan pengolahan data yang terdapat pada sistem dengan mengisi form login terlebih dahulu, jika salah maka akan sistem akan melaukan penolakan dan admin mengulangi proses pengisisan form login. Jika benar maka admin akan masuk kedalam menu home kemudian admin dapat melakukan pengolahan data yang ingin dilakukan.

* 1. Activity Diagram Pengolahan Grafik Data

Activity Diagram ini menjelaskan bagaimana alur tentang pengolahan grafik yang terjadi pada sistem *data warehouse* ini, pengolahan proses tersebut dilakukan oleh admin didalam sistem. Untuk lebih jelas dapat diligat pada gamabr 4.29 dibawah ini sebagai berikut :

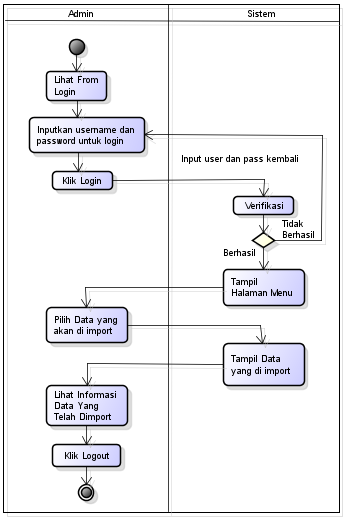


**Gambar 4.29 Activity Diagram Pengolahan Grafik Data**

Pada gambar 4.29 menggambarkan interaksi aktor pada sistem yang akan dilakukan oleh aktor admin dalam melakukan pengolahan grafik data yang terdapat pada sistem dengan mengisi form login terlebih dahulu, jika salah maka akan sistem akan melaukan penolakan dan admin mengulangi proses pengisisan form login. Jika benar maka admin akan masuk kedalam menu home kemudian admin memilihan data untuk melakukan pengolahan grafik data yang ingin dilakukan.

* 1. Activity Diagram Import Data

Activity Diagram dalam import data menjelaskan tentang alur dalam proses pengimportan data yang dilakukan oleh admin. Untuk lebih jelasnya bisa dilihat pada gambar 4.30 berikut dibawah ini :

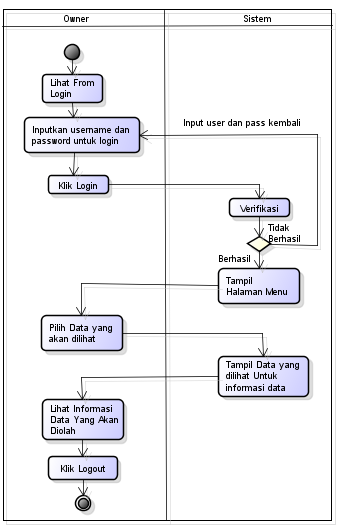


**Gambar 4.30 Activity Diagram Import Data**

Pada gambar 4.30 menggambarkan interaksi aktor pada sistem yang akan dilakukan oleh aktor admin dalam melakukan pengolahan data yang terdapat pada sistem dengan mengisi form login terlebih dahulu, jika salah maka akan sistem akan melaukan penolakan dan admin mengulangi proses pengisisan form login. Jika benar maka admin akan masuk kedalam menu home kemudian admin dapat melakukan pengolahan dalam mengimport data.

* 1. Activity Diagram Pengolahan Data Pada Owner

*Activity Diagram* pengolahan data pada owner yaitu menjelaskan tentang kegiatan yang dilakukan oleh owner pada sistem. Dapat dilihat pada gambar dibawah ini :

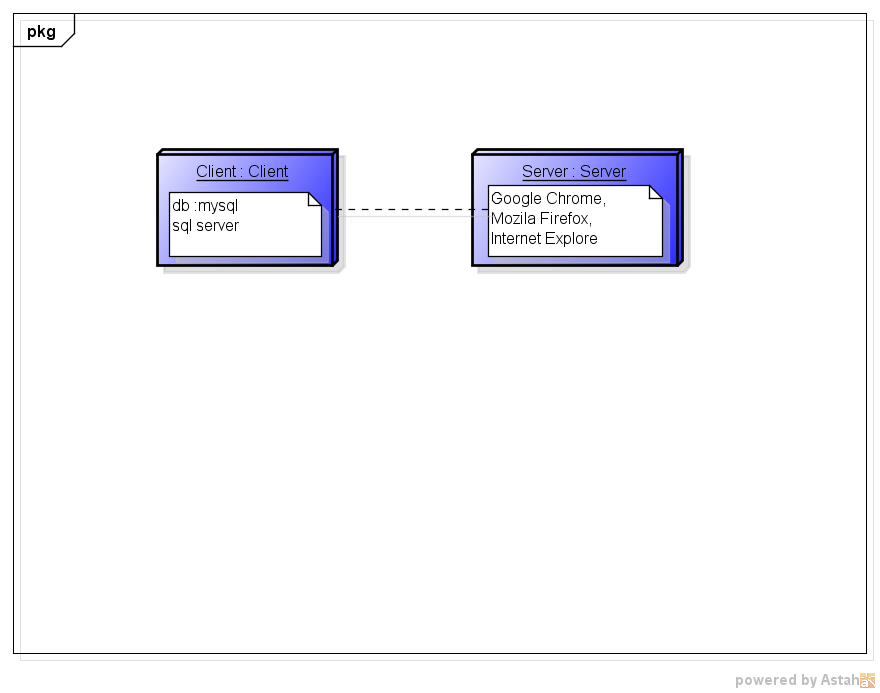


**Gambar 4.31 Activity Diagram Pengolahan Data Pada Owner**

Pada gambar 4.31 menggambarkan interaksi aktor pada sistem yang akan dilakukan oleh aktor owner dalam melakukan pengolahan data yang terdapat pada sistem dengan mengisi form login terlebih dahulu, jika salah maka akan sistem akan melaukan penolakan dan owner mengulangi proses pengisisan form login. Jika benar maka owner akan masuk kedalam menu home kemudian admin dapat melakukan pengolahan data yang ingin dilakukan.

# Deployment Diagram

Deployment diagram menggambarkan secara lengkap bagaimana tata letak dan komponen di *deploy* dalam infrastruktur sistem. Kemampuan jaringan pada kondisi tertentu, spesifikasi server, dan lain-lain yang bersifat fisikal. Dengan cara memperlihatkan bagian *software* yang berjalan pada bagian *hardware*. Gambaran deployment diagram dapat dilihat pada gambar 4.32 dibawah ini :



**Gambar 4.32 Deployment Diagram**

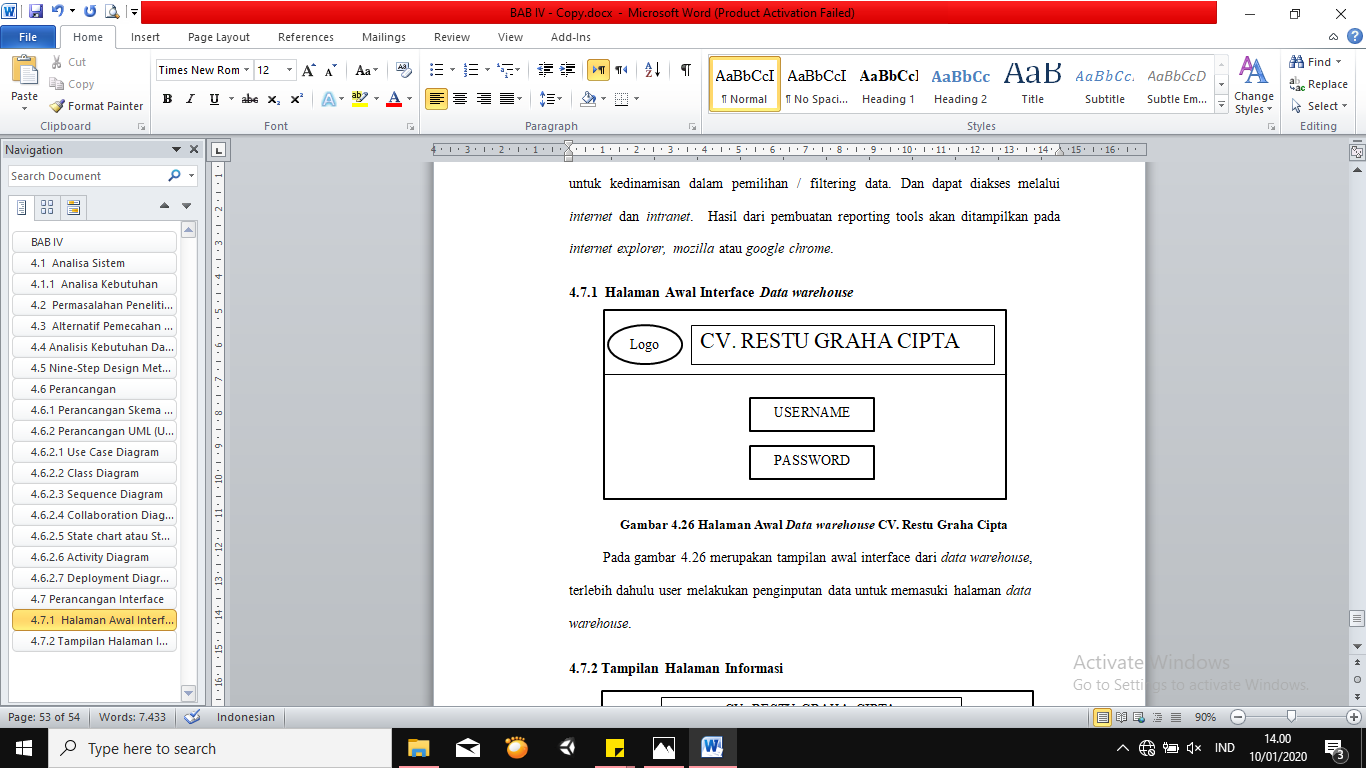
Pada Gambar 4.32 deployment diagram *data warehouse* terdiri dari client dan server. Pada client terdapat database MySql dan SQL Server. Sedangkan pada server terdapat Google Chrome, Mozila Firefox dan Internet Explore yang berfungsi untuk mempermudah dalam proses penggunaan *data warehouse* atau sebagai koneksi penghubung antar pengguna dan *data warehouse*.

# Perancangan Interface

Perancangan interface merupakan perancangan sistem untuk memberitahukan tampilan sistem yang digunakan dalam berinteraksi dengan pengguna. Perancangan ini menggunakan bantuan *tool SQL Server Reporting Service*. Hal ini bertujuan untuk kedinamisan dalam pemilihan / filtering data. Dan dapat diakses melalui *internet* dan *intranet*. Hasil dari pembuatan reporting tools akan ditampilkan pada *internet explorer, mozilla* atau *google chrome*.

# Tampilan Interface

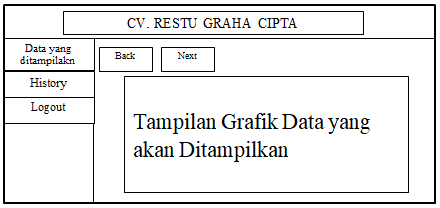
Perancangan interface adalah perancangan index yang ditampilkan pada user saat mengunjungi *website data warehouse*. Pada tampilan awal program atau interface ini menampilkan form header, login yang terlihat pada gambar 4.33 sebagai berikut :



**Gambar 4.33 Halaman Awal *Data warehouse* CV. Restu Graha Cipta**

Pada gambar 4.33 merupakan tampilan awal interface dari *data warehouse*, terlebih dahulu user melakukan penginputan data untuk memasuki halaman *data warehouse*.

# Tampilan Halaman Informasi



**Gambar 4.34 Tampilan Halaman Informasi**

Pada gambar 4.34 halaman informasi merupakan informasi tentang kondisi data yang dapat dilihat sesuai keinginan. Pada menu utama tombol klik berfungsi untuk memilih data yang ingin ditampilkan baik berdasarkan tahun, daerah, lokasi, tenaga ahli dan lain-lainnya. Adanya tampilan grafik untuk mengetahui perkembangan data yang diinginkan sebagai informasi dalam pengambilan sebuah keputusan.

**BAB V**

**IMPLEMENTASI DAN PENGUJIAN**

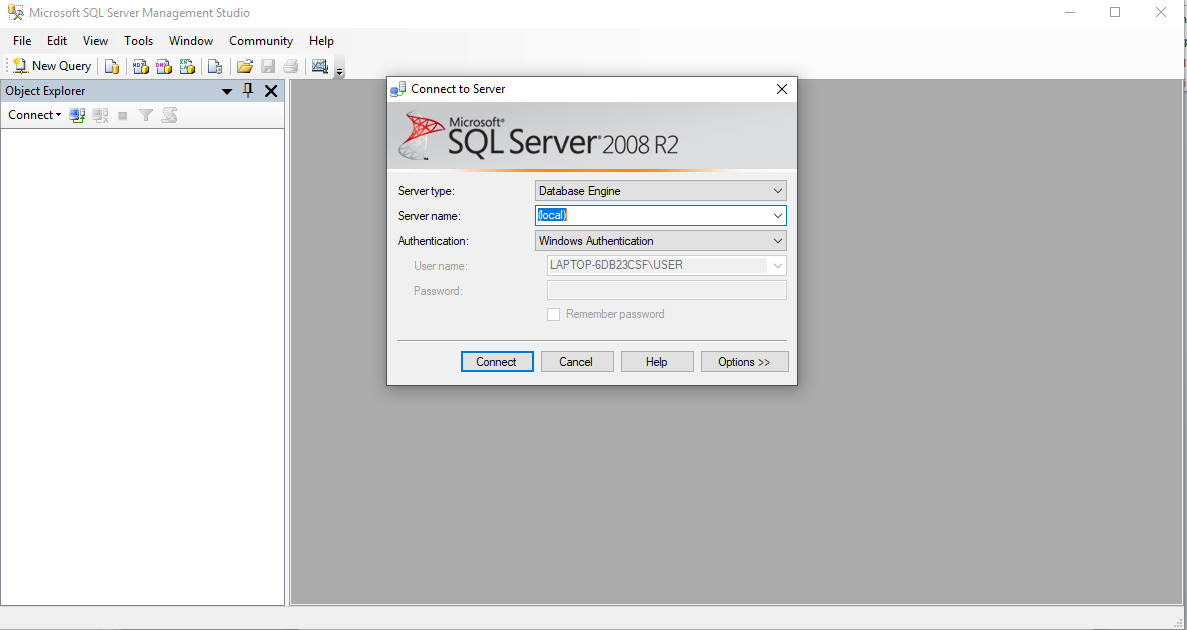
# 5.1 Implementasi

Implementasi *data warehouse* ini dirancang dalam melakukan *import* data mentah ke dalam *database* yang ada di *SQL Server 2008 R2* sampai pada proses analisis data.

# Implementasi Integration Service

SQL Server Integration Services (SSIS) merupakan proses *data warehouse* menggunakan proses ETL (*Extract, Transform and Loading*). *Extract* yaitu untuk pengumpulan data dari berbagai sumber, *transform* merupakan proses pembersihan data dan *loading* merupakan proses penyimpanan data kedalam database. Berikut ini adalah langkah-langkah yang dapat dilakukan pada *integration service*:

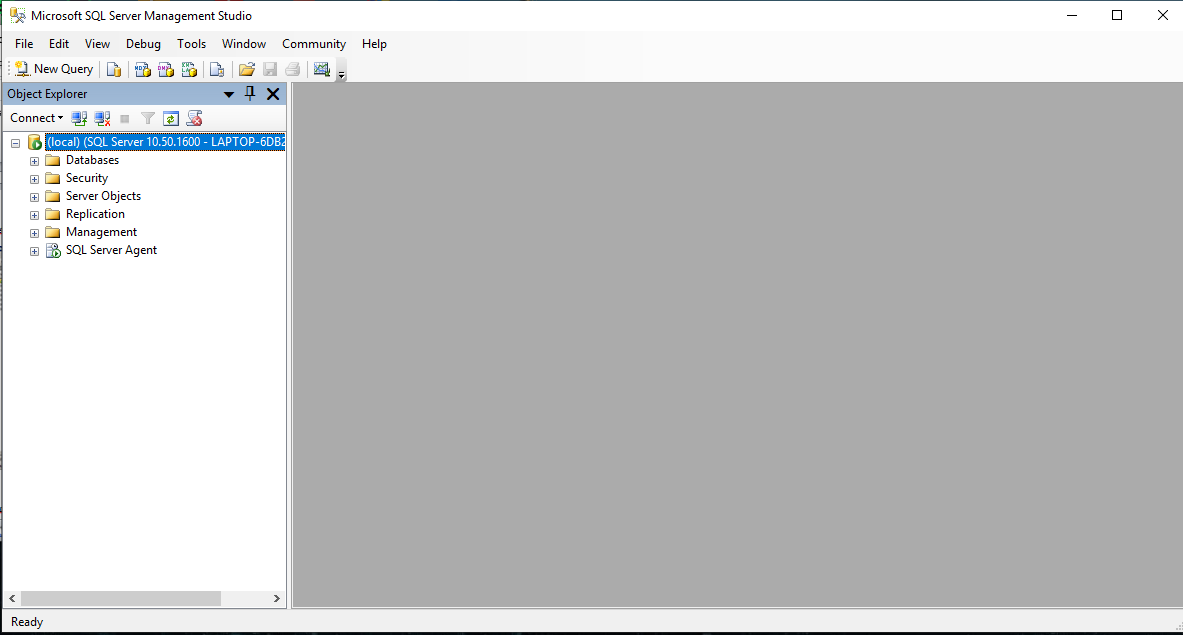
Langkah pertama yang dilakukan yaitu membuka program *SQL Server Management Studio* dan melakukan koneksi ke *server*. Klik *Connect* pada *button* tersebut. Berikut ini merupakan gambar 5.1 sebagai berikut:

****

**Gambar 5.1 *Connect to Server***

Pada gambar 5.1 merupakan tampilan awal dari *SQL Server Manajement*, akan muncul tampilan *connect to server* yang terdiri dari *server type, server name* dan *authentication* dan klik OK untuk masuk kedalam *SQL Server Manajement*.

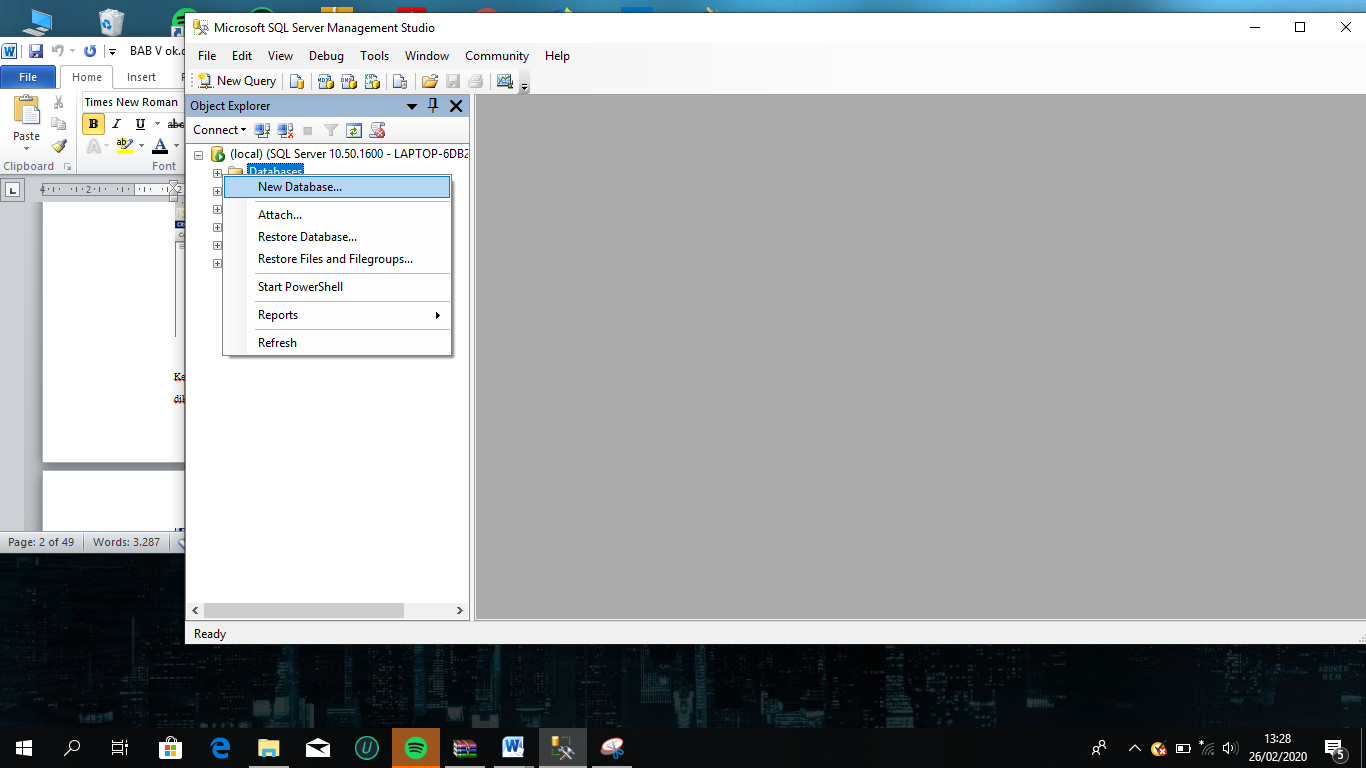
Setelah klik tombol *connect* pada tampilan *connect to server* maka akan muncul tampilan gambar 5.2 dibawah ini sebagai berikut :



**Gambar 5.2 Tampilan setelah *Connect***

Pada gambar 5.2 merupakan tampilan yang akan muncul setelah *connect* kedalam *SQL Server Manajement* yang terdiri dari *database, security, server objects, replication* dan *management*.

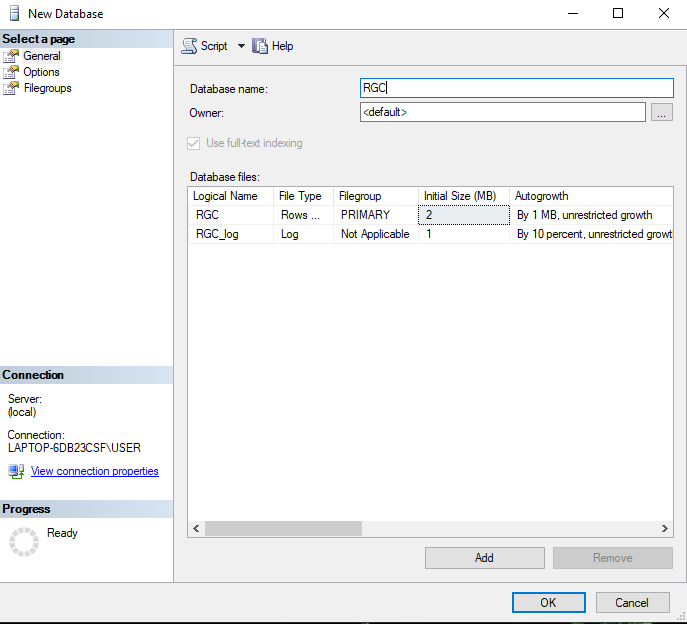
Langkah selanjutnya membuat *database* untuk menampung data pada perusahaan CV. Restu Graha Cipta. Caranya klik kanan pada *database* kemudian klik *new database* seperti gambar 5.3 dibawah ini :



**Gambar 5.3 Membuat *Database***

Pada gambar 5.3 pembuatan *database*, pada pembuatan *database* klik kanan untuk membuat *database* baru.

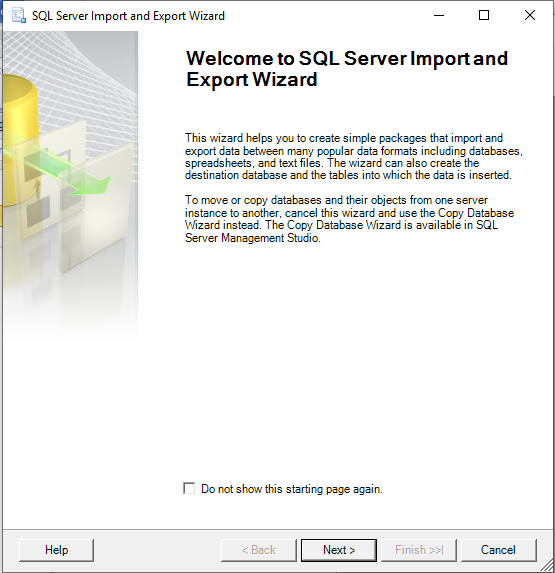
Setelah klik kanan, akan muncul kotak menu *New Database*. Pada *database name* buatlah dengan nama *database* RGC, lalu klik OK seperti gambar 5.4 sebagai berikut:



**Gambar 5.4 *New Database***

Pada gambar 5.4 merupakan tampilan pembuatan *database* baru. Buat *database* dengan nama RGC, perhatikan *local name*, *file type*,  *file group*, dan *intia*l *size* pada database.

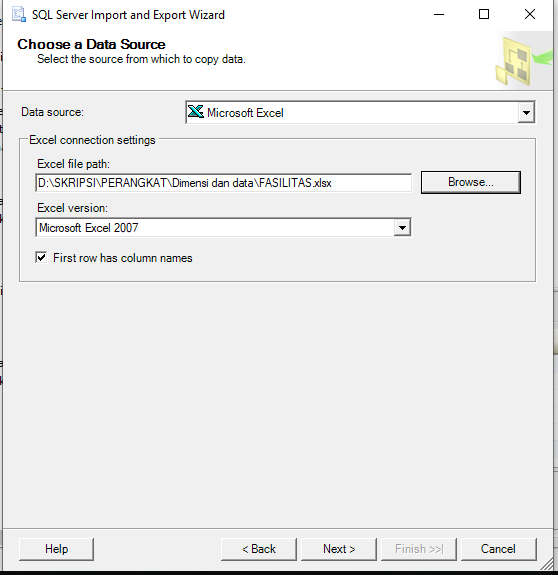
Kemudian lakukan *import* data fasilitas ke dalam *database* klik kanan pada *database* RGC, pilih *task*, kemudian klik *import data*, setelah itu akan muncul tampilan gambar 5.5 sebagai berikut ini :



**Gambar 5.5 *SQL Server Import* dan *Export Wizard* Fasilitas**

Pada gambar 5.5 merupakan tampilan *SQL Server Import and Export Wizard*. Klik *next* untuk melakukan tahapan selanjutnya.

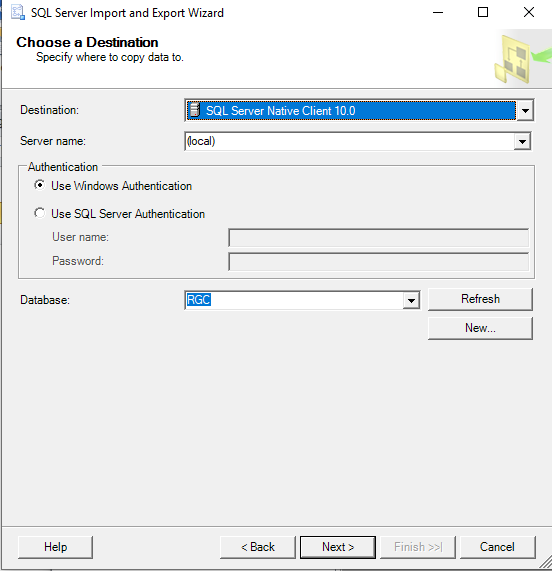
Klik *next*, maka akan muncul *Choose a Data Source*. Tahap ini berfungsi untuk memilih data yang akan di *import* ke dalam *database*. Berikut tampilan pada gambar 5.6 dibawah ini :



**Gambar 5.6 *Choose a* Data Source Fasilitas**

Pada gambar 5.6 merupakan tampilan dari *choose a data source*. Data yang terdapat pada perusahaan memiliki format *Microsoft Excel* maka select *data Source* yang akan dipilih yaitu *Microsoft Excel*. Lalu klik *browse* untuk mencari data dan klik *next*.

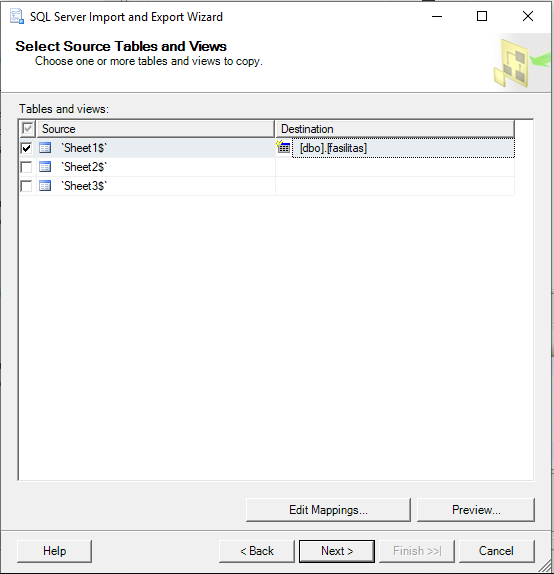
Kemudian akan muncul *menu Choose a Destination*. Tahap ini berfungsi untuk memilih data *source* yang akan di *copy*. Pada *Destination* pilih *SQL Native Client*. Kemudian pada *server* dan *database* dipilih *database RGC*, dan klik next. Berikut tampilan seperti gambar 5.7 dibawah ini :



**Gambar 5.7 *Choose a Destination* Fasilitas**

Pada gambar 5.7 merupakan tampilan dari *choose a destination*. Pada *destination* pilih *SQL Server Native Client 10.0* dengan server name (*local*) dan authentication select *Use Windows Authentication*. Pada database select berdasarkan database yang digunakan dalam penyimpanan data.

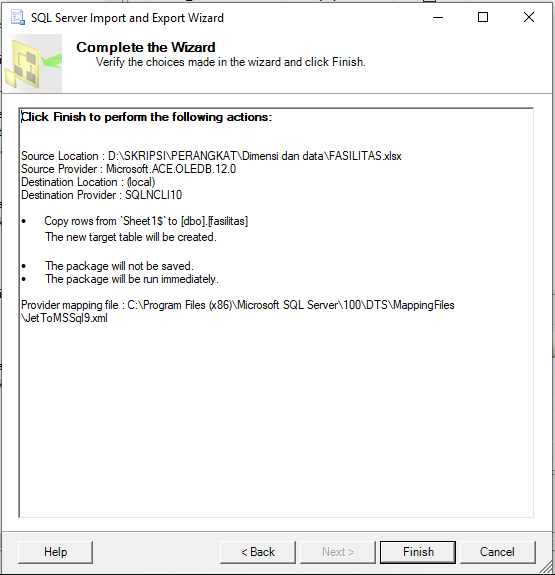
Kemudian akan muncul *Select Source Tables and View*. Berfungsi untuk memilih pada *table* mana yang akan digunakan pada *Microsoft Excel*. Lalu pilih *table* yang akan digunakan dengan *check list tabel* pada data dan *klik next*. Berikut tampilan seperti gambar 5.8 dibawah ini :



**Gambar 5.8 *Select Source Tables and Views* Fasilitas**

Pada gambar 5.8 merupakan *select source tables* and *views* dari data *excel* yang telah dipilih. *Source table* and *views* terdiri dari beberapa sheet, pilih sheet data yang dibutuhkan. Pada *destination* ubah nama fasilitas untuk pemanggilan tabel pada *database*.

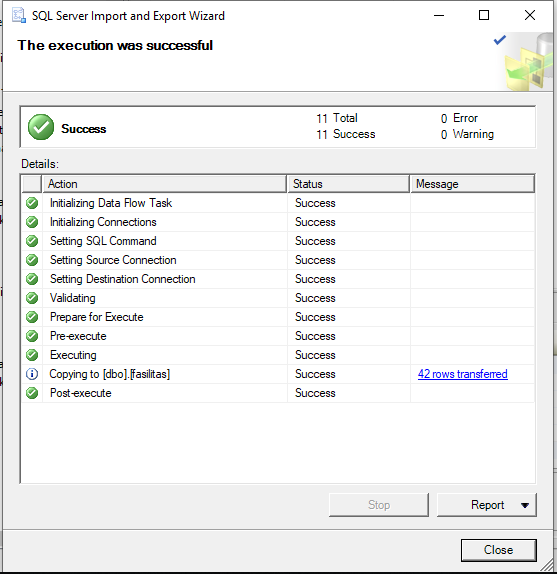
Maka akan muncul *menu Complete the Wizard*. Klik *finish* lalu data akan di *import* secara otomatis kedalam *database* RGC. Berikut tampilan seperti gambar 5.9 dibawah ini :



**Gambar 5.9 Complete The Wizard Fasilitas**

Pada gambar 5.9 tampilan *complete wizard* yaitu berupa rincian data yang telah dipilih dan memastikan data yang dipilih dengan benar, dan klik *finish* jika data sudah benar.

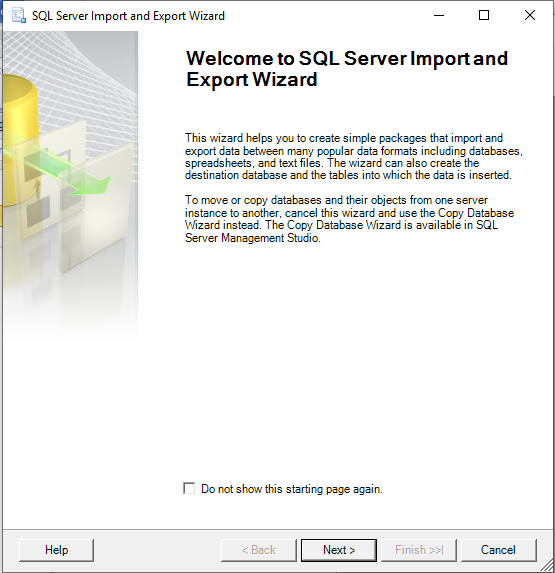
Berikut ini tampilan sukses dalam proses *import* data ke *database* seperti gambar 5.10 dibawah ini:



**Gambar 5.10 *Success Import Data* Fasilitas**

Pada gambar 5.10 merupakan tampilan sukses untuk mengimport data fasilitas kedalam *database* RGC.

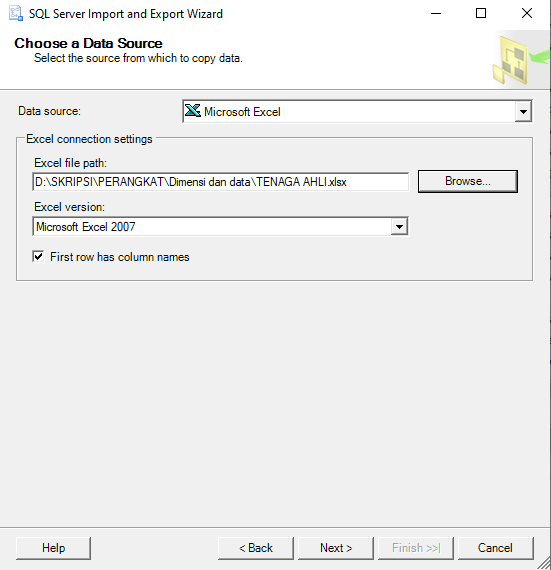
Untuk Import data tenaga ahli ke dalam *database* seperti langkah diatas dengan klik kanan pada *database* RGC, pilih *task*, kemudian klik *import* data, setelah itu akan muncul gambar 5.11 sebagai berikut ini :



**Gambar 5.11 *SQL Server Import* dan *Export Wizard* Tenaga Ahli**

Pada gambar 5.11 merupakan tampilan *SQL Server Import and Export Wizard*. Klik *next* untuk melakukan tahapan selanjutnya.

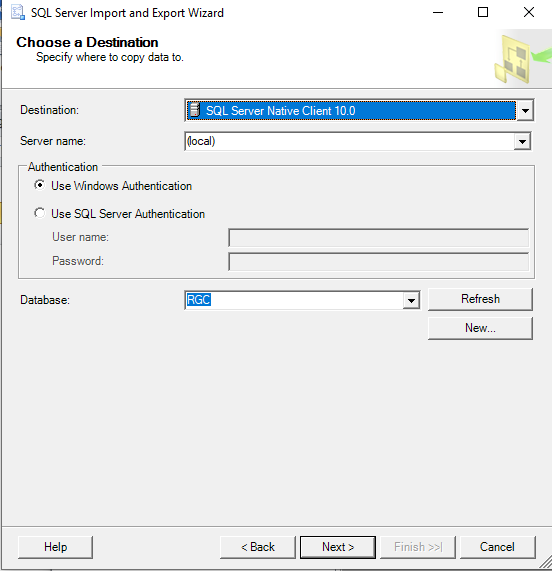
Klik *next*, maka akan muncul *Choose a Data Source*. Tahap ini berfungsi untuk memilih data yang akan di *import* ke dalam *database*. Berikut tampilan seperti gambar 5.12 dibawah ini :



**Gambar 5.12 *Choose a Data Source* Tenaga Ahli**

Pada gambar 5.12 merupakan tampilan dari *choose a data source*. Data yang terdapat pada perusahaan memiliki format *Microsoft Excel* maka select *data Source* yang akan dipilih yaitu *Microsoft Excel*. Lalu klik *browse* untuk mencari data dan klik *next*.

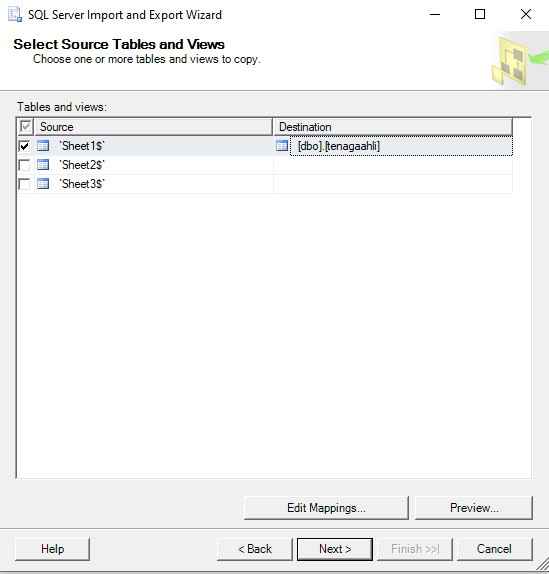
Kemudian akan muncul *menu Choose a Destination*. Tahap ini berfungsi untuk memilih data *source* yang akan di *copy*. Pada *Destination* pilih *SQL Native Client*. Kemudian pada *server* dan *database* dipilih *database RGC*, dan klik next. Berikut ini seperti gambar 5.13 dibawah ini :



**Gambar 5.13 *Choose a Destination* Tenaga Ahli**

Pada gambar 5.13 merupakan tampilan dari *choose a destination*. Pada *destination* pilih *SQL Server Native Client 10.0* dengan server name (*local*) dan authentication select *Use Windows Authentication*. Pada database select berdasarkan database yang digunakan dalam penyimpanan data.

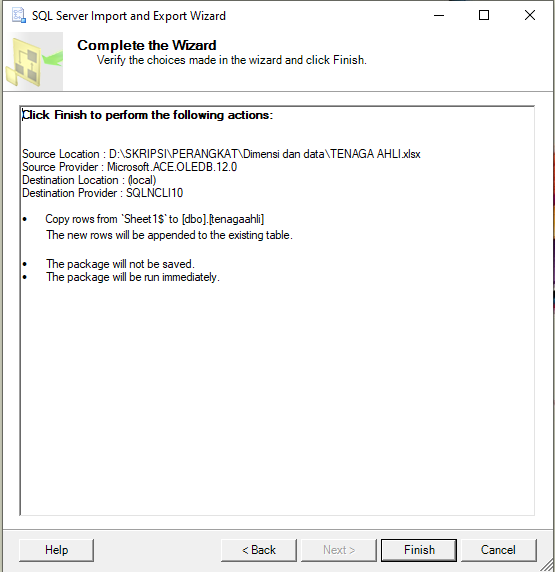
Kemudian akan muncul *Select Source Tables and View*. Berfungsi untuk memilih pada *table* mana yang akan digunakan pada *Microsoft Excel*. Lalu pilih *table* yang akan digunakan dengan *check list tabel* pada data dan *klik next*. Berikut tampilan seperti gambar 5.14 dibawah ini :



**Gambar 5.14 *Select Source Tables and Views* Tenaga Ahli**

Pada gambar 5.14 merupakan *select source tables* and *views* dari data *excel* yang telah dipilih. Pada *destination* ubah nama fasilitas untuk pemanggilan tabel pada *database*.

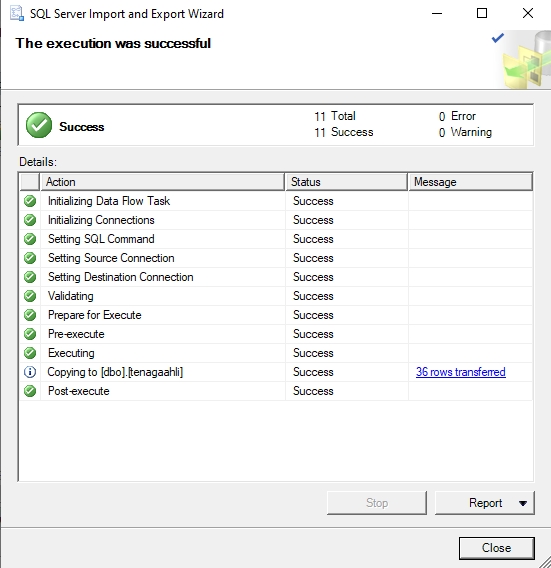
Maka akan muncul *menu Complete the Wizard*. Klik *finish* lalu data akan di *import* secara otomatis kedalam *database* RGC. Berikut tampilan seperti gambar 5.15 dibawah ini



**Gambar 5.15 Complete The Wizard Tenaga Ahli**

Pada gambar 5.15 tampilan *complete wizard* memastikan data yang dipilih dengan benar, dan klik *finish* jika data sudah benar.

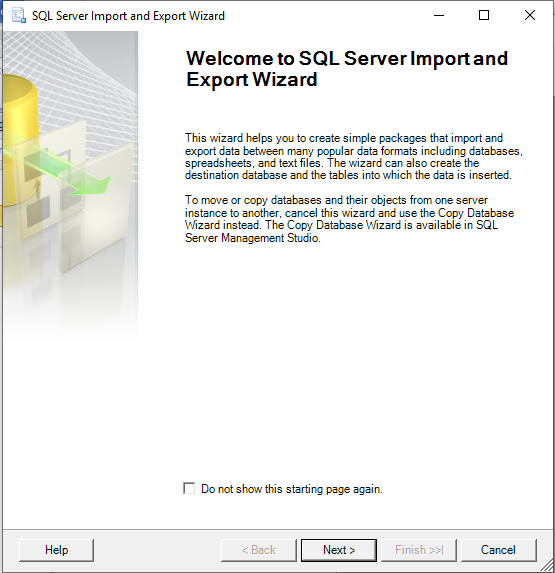
Berikut ini tampilan sukses dalam proses *import* data ke *database* seperi gambar 5.16 dibawah ini:



**Gambar 5.16 *Success Import Data* Tenaga Ahli**

Pada gambar 5.16 merupakan tampilan sukses untuk mengimport data tenaga ahli kedalam *database* RGC.

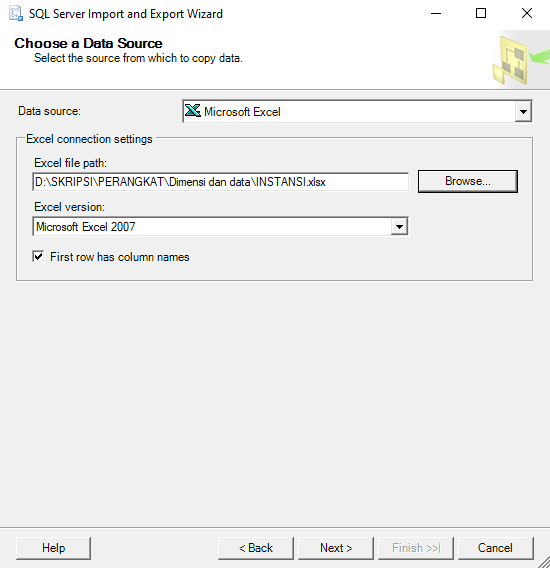
Lakukan langkah yang sama untuk *import* data instansi ke dalam *database* dengan mengklik kanan pada *database* RGC, pilih *task,* kemudian klik *import data*, setelah itu akan muncul tampilan gambar 5.17 sebagai berikut ini :



**Gambar 5.17 *SQL Server Import* dan *Export Wizard* Instansi**

Pada gambar 5.17 tampilan *complete wizard* yaitu berupa rincian data yang telah dipilih dan memastikan data yang dipilih dengan benar, dan klik *finish* jika data sudah benar.

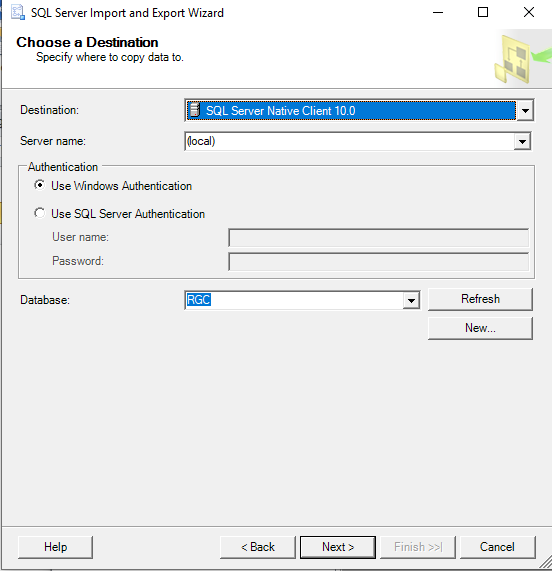
Klik *next*, maka akan muncul *Choose a Data Source*. Tahap ini berfungsi untuk memilih data yang akan di *import* ke dalam *database*. Berikut tampilan seperti gambar 5.18 dibawah ini :



**Gambar 5.18 *Choose a Data Source* Tenaga Ahli**

Pada gambar 5.18 pada data yang terdapat pada perusahaan memiliki format *Microsoft Excel*, maka pada *Data Source* yang dipilih *yaitu Microsoft Excel*. Lalu klik *browse* untuk mencari data dan klik *next*.

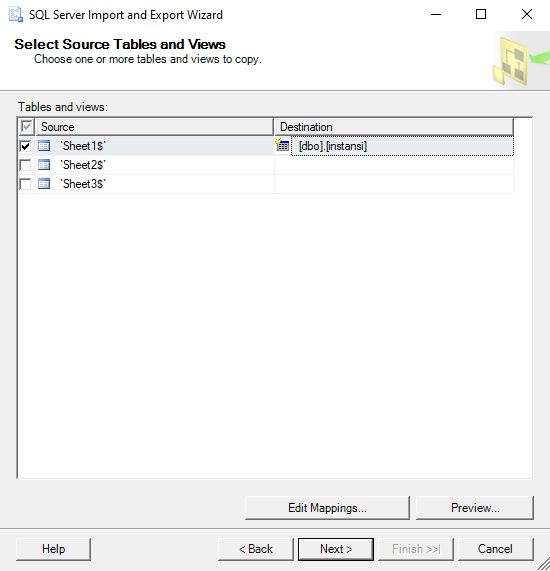
Kemudian akan muncul *menu Choose a Destination*. Tahap ini berfungsi untuk memilih data *source* yang akan di *copy*. Pada *Destination* pilih *SQL Native Client*. Kemudian pada *server* dan *database* dipilih *database RGC*, dan klik next. Berikut tampilan seperti gambar 5.19 dibawah ini



**Gambar 5.19 *Choose a Destination* Instansi**

Pada gambar 5.19 merupakan tampilan dari *choose a destination*. Pada *destination* pilih *SQL Server Native Client 10.0* dengan server name (*local*) dan authentication select *Use Windows Authentication*. Pada database select berdasarkan database yang digunakan dalam penyimpanan data.

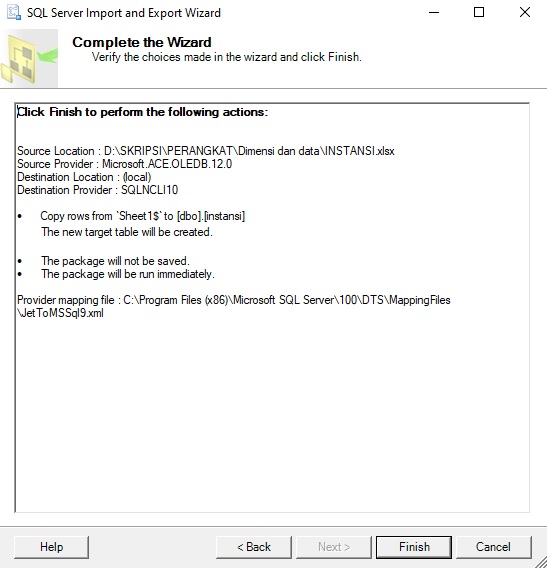
Kemudian akan muncul *Select Source Tables and View*. Berfungsi untuk memilih pada *table* mana yang akan digunakan pada *Microsoft Excel*. Lalu pilih *table* yang akan digunakan dengan *check list tabel* pada data dan *klik next*. Berikut tampilan seperti gambar 5.20 dibawah ini



**Gambar 5.20 *Select Source Tables and Views* Instansi**

Pada gambar 5.20 merupakan *select source tables* and *views* dari data *excel* yang telah dipilih. Pada *destination* ubah nama fasilitas untuk pemanggilan tabel pada *database*.

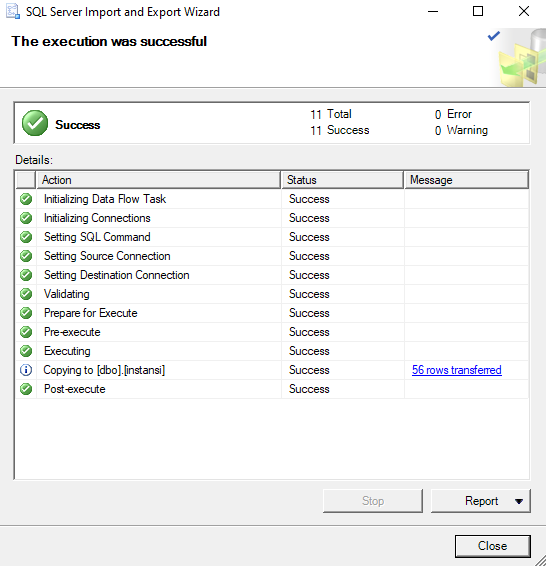
Maka akan muncul *menu Complete the Wizard*. Klik *finish* lalu data akan di *import* secara otomatis kedalam *database* RGC. Berikut tampilan seperti gambar 5.21 dibawah ini



**Gambar 5.21 Complete The Wizard Instansi**

Pada gambar 5.21 tampilan *complete wizard* memastikan data yang dipilih dengan benar, dan klik *finish* jika data sudah benar.

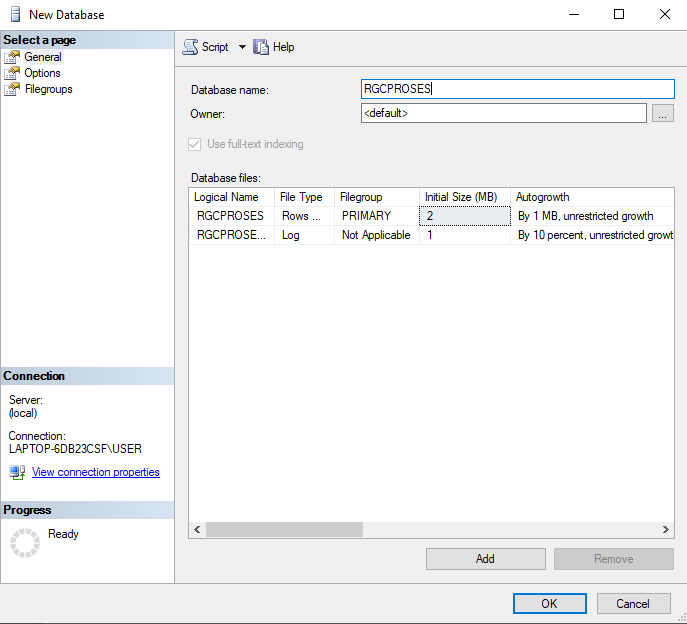
Berikut ini tampilan sukses dalam proses *import* data ke *database* seperti gambar 5.22 dibawah ini.



**Gambar 5.22 *Success Import Data* Instansi**

Pada gambar 5.22 merupakan tampilan sukses untuk mengimport data instansi kedalam *database* RGC.

Langkah selanjutnya pembuatan *database* untuk proses yang berisi semua tabel dimensi, tabel fakta, dan tabel proses yang akan digunakan dalam membuat *data warehouse*. Pada langkah ini adanya pembuatan database baru untuk menampung semua tabel dimensi, tabel fakta dan tabel proses yang data tersebut berasal dari database sebelumnya. Langkah awal yang dilakukan klik kanan *database*, klik *new database*. Kemudian pada *database name* ketik RGCPROSES, lalu klik OK. Berikut tampilan seperti gambar 5.23 dibawah ini



**Gambar 5.23 Membuat *Database* Proses**

Pada gambar 5.23 merupakan tampilan pembuatan *database* baru. Buat *database* dengan nama RGCPROSES, perhatikan *local name*, *file type*,  *file group*, dan *intia*l *size* pada *database*.

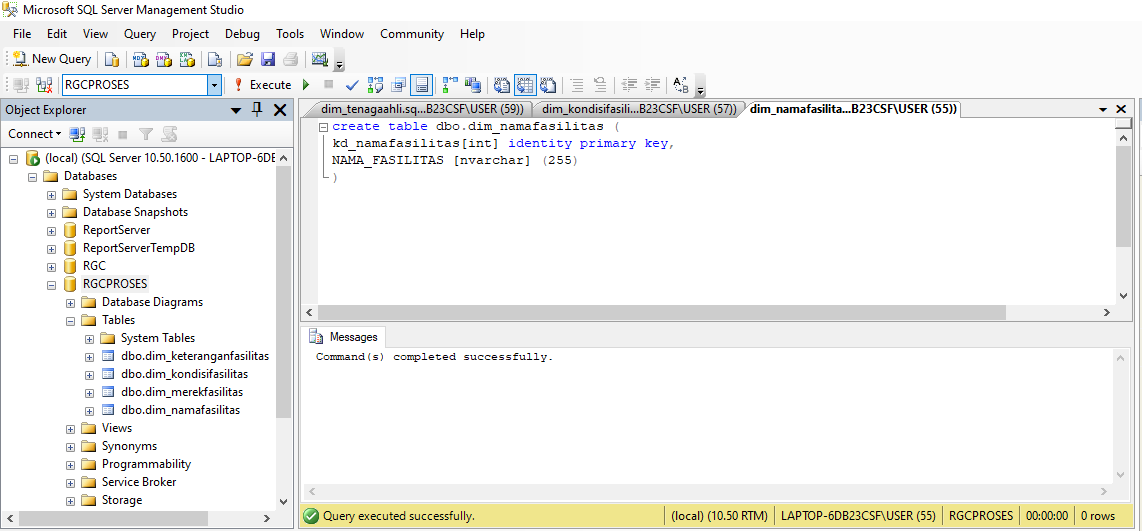
Setelah *database* RGCPROSES terbentuk, maka langkah selanjutnya adalah membuat seluruh tabel dimensi *data warehouse*. Caranya adalah klik kanan pada RGCPROSES, klik *New Query*. Lalu ketikan *Query* berikut untuk membuat dim\_namafasilitas:

create table dbo.dim\_namafasilitas (

kd\_namafasilitas [int] identity primary key,

NAMA\_FASILITAS [nvarchar] (255))

Kemudian klik *execute*. Jika berhasil maka akan tampil gambar 5.24 sebagai berikut.



**Gambar 5.24 Membuat Dimensi Nama Fasilitas**

Pada gambar 5.24 merupakan query dari dimensi nama fasilitas.

Untuk membuat tabel-tabel dimensi yang lain caranya sama dengan dim\_namafasilitas. berikut *query* nya:

1. *Query* dim\_merekfasilitas:

create table dbo.dim\_merekfasilitas (

kd\_merek [int] identity primary key,

MEREK\_FASILITAS [nvarchar] (255))

1. *Query* dim\_jenisfasilitas:

create table dbo.dim\_jenisfasilitas (

kd\_jenis [int] identity primary key,

JENIS\_FASILITAS [nvarchar] (255))

1. *Query* dim\_kondisifasilitas:

create table dbo.dim\_kondisifasilitas (

kd\_kondisi [int] identity primary key,

KONDISI\_FASILITAS [nvarchar] (255))

1. *Query* dim\_keteranganfasilitas:

create table dbo.dim\_keteranganfasilitas (

kd\_keterangan [int] identity primary key,

KETERANGAN\_FASILITAS [nvarchar] (255))

1. *Query* dim\_tenagaahli:

create table dbo.dim\_tenagaahli (

id\_tenagaahli [int] identity primary key,

TENAGA\_AHLI [nvarchar] (255))

1. *Query* dim\_jurusan:

create table dbo.dim\_jurusan (

kd\_jurusan [int] identity primary key,

JURUSAN [nvarchar] (255))

1. *Query* dim\_universitas:

create table dbo.dim\_universitas (

kd\_universitas [int] identity primary key,

UNIVERSITAS [nvarchar] (255))

1. *Query* dim\_tingkatpendidikan:

create table dbo.dim\_tingkatpendidikan (

kd\_tingkatpendidikan [int] identity primary key,

TINGKAT\_PENDIDIKAN [nvarchar] (255))

1. *Query* dim\_status:

create table dbo.dim\_status (

kd\_status [int] identity primary key,

STATUS [nvarchar] (255))

1. *Query* dim\_subbidang:

create table dbo.dim\_subbidang (

kd\_subbidang [int] identity primary key,

SUB\_BIDANG [nvarchar] (255))

1. *Query* dim\_instansi:

create table dbo.dim\_instansi (

kd\_instansi [int] identity primary key,

INSTANSI [nvarchar] (255))

1. *Query* dim\_waktu:

create table dbo.dim\_waktu (

kd\_waktu [int] identity primary key,

TANGGAL [nvarchar] (255),

HARI [datetime] ,

BULAN [nvarchar] (255),

TAHUN [nvarchar] (255))

1. *Query* dim\_pemberijasa:

create table dbo.dim\_pemberijasa (

id\_pemberijasa [int] identity primary key,

PEMBERI\_JASA [nvarchar] (255),

ALAMAT\_PEMBERIJASA [nvarchar] (255))

Pada dimensi ibu kota, provinsi, kabupaten dan daerah memiliki *foreign key* yang saling terhubung satu sama lain, berikut *query* dibawah ini :

1. *Query* dim\_ibukota

create table dbo.dim\_ibukota (

kd\_ibukota [int] identity primary key,

IBUKOTA [nvarchar] (255))

1. *Query* dim\_provinsi

create table dbo.dim\_provinsi (

kd\_provinsi [int] identity primary key,

kd\_ibukota [int] foreign key references dim\_ibukota (kd\_ibukota),

PROVINSI [nvarchar] (255))

1. *Query* dim\_kabupaten

create table dbo.dim\_kabupaten (

kd\_kabupaten [int] identity primary key,

kd\_provinsi [int] foreign key references dim\_provinsi (kd\_provinsi),

KABUPATEN [nvarchar] (255))

1. *Query* dim\_daerah

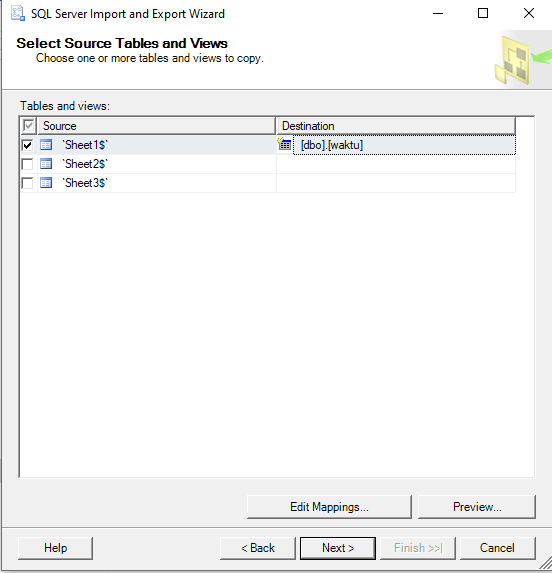
create table dbo.dim\_daerah (

kd\_daerah [int] identity primary key,

kd\_kabupaten [int] foreign key references dim\_kabupaten (kd\_kabupaten),

DAERAH [nvarchar] (255))

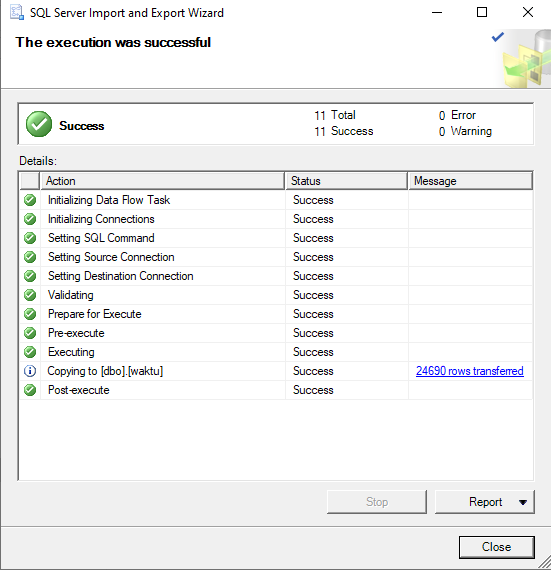
Langkah selanjutnya adalah membuat dim\_waktu. Caranya adalah klik kanan RGCPROSES, pilih *task*, lalu klik *import data*. Lalu pada *data Source* pilih *Microsoft Excel*, lalu *browse* untuk mengambil data waktu lalu klik *next*. Kemudian mucul *Choose a Destination* pada *destination* pilih *SQL Native Client*, kemudian *database* pilih db\_proses, lalu klik *next*. Kemudian akan muncul *Select Source Tables and Views*. Pilih *Sheet1* kerena data waktu berada pada *sheet1* pada *Microsoft Excel*. Lalu klik *next*. Berikut tampilan seperti gambar 5.25 dibawah ini

****

**Gambar 5.25 *Select Source Tables and Views* dim\_waktu**

Pada gambar 5.26 merupakan *select source tables* and *views* dari data *excel* yang telah dipilih. Pada *destination* ubah nama fasilitas untuk pemanggilan tabel pada *database*.

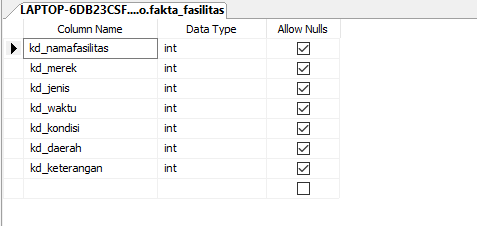
Kemudian akan tampil *Complete the Wizard* sebagai berikut. Klik *Finish* seperti gambar 5.26 dibawah ini :

****

**Gambar 5.26 *Complete the Wizard* dim\_waktu**

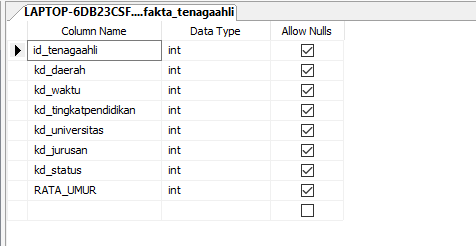
Pada gambar 5.26 merupakan tampilan sukses untuk mengimport data dimensi waktu kedalam *database* RGC.

Langkah selanjutnya adalah membuat tabel fakta. Tabel fakta ini berisi *primary key* dari masing-masing dimensi dan *measure*. Caranya adalah klik kanan menu *table* pada RGCPROSES, klik *new table*. Lalu ketik *column name* dan *data type* seperti gambar berikut. Kemudian klik kanan pada dbo.Table\_1, pilih *save* Tabel\_1. Kemudian akan muncul kotak perintah untuk memberikan nama tabel. Ketikan fakta\_fasilitas, fakta\_tenagaahli dan fakta\_instansi, lalu klik OK.



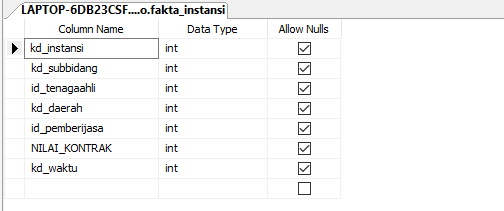
**Gambar 5.27 Pembuatan isi tabel Fakta Fasilitas**

Pada gambar 5.27 merupakan tabel fakta fasilitas yang ada pada database RGCPROSES.



**Gambar 5.28 Pembuatan isi tabel Fakta Tenaga Ahli**

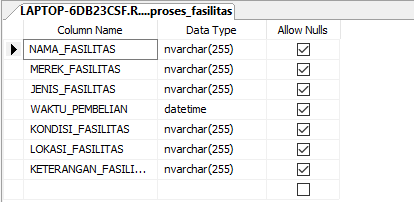
Pada gambar 5.28 merupakan tabel fakta tenaga ahli yang ada pada database RGCPROSES.



**Gambar 5.29 Pembuatan isi tabel Fakta Instansi**

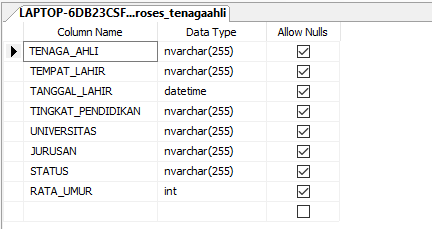
Pada gambar 5.29 merupakan tabel fakta instansi yang ada pada database RGCPROSES.

Langkah selanjutnya adalah membuat tabel proses untuk fasilitas tenaga ahli dan instansi. Tabel ini berfungsi menggabungkan data-data yang ada didalam tabel-tabel dimensi. Yang kemudian dihubungkan ke tabel fakta, sehingga tabel fakta tersebut akan berisi *key-key* saja. Caranya adalah kanan menu tabel pada RGCPROSES, pilih *new* tabel, kemudian ketikan column name dan data *type* nya seperti gambar berikut. Kemudian klik kanan pada *dbo.Table\_1*, pilih *save table\_1*. Maka akan muncul perintah untuk memberikan nama tabel proses lalu klik OK, seperti gambar berikut:



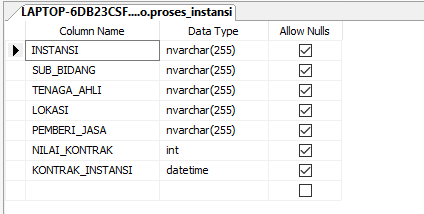
**Gambar 5.30 Pembuatan Tabel Proses Fasilitas**

Pada gambar 5.30 merupakan tabel dari proses fasilitas untuk menghubungkan data pada tabel fakta fasilitas yang berisi key-key saja.



**Gambar 5.31 Pembuatan Tabel Proses Tenaga Ahli**

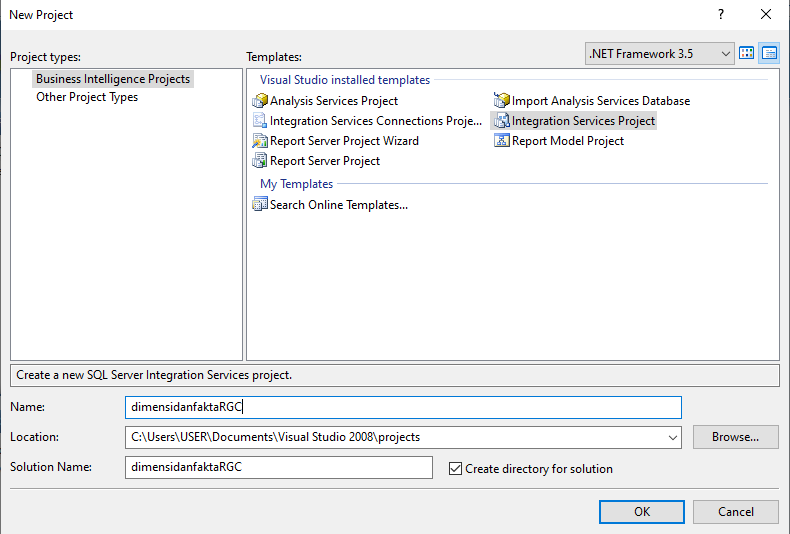
Pada gambar 5.31 merupakan tabel dari proses tenaga ahli untuk menghubungkan data pada tabel fakta tenaga ahli yang berisi key-key saja.



**Gambar 5.32 Pembuatan Tabel Proses Instansi**

Pada gambar 5.31 merupakan tabel dari proses instansi untuk menghubungkan data pada tabel fakta instansi yang berisi key-key saja.

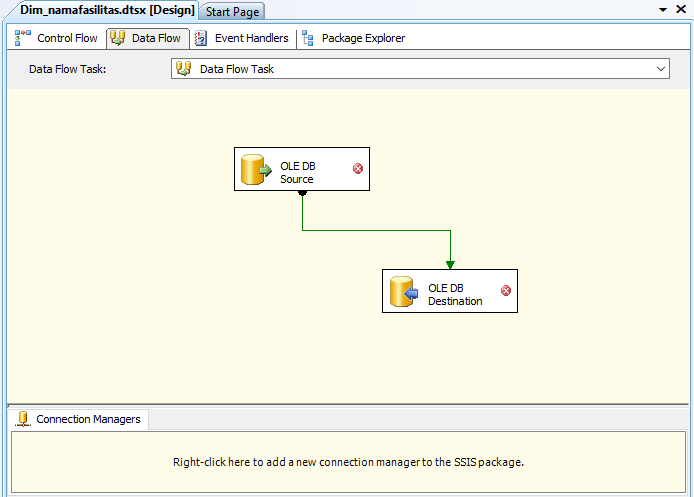
Langkah selanjutnya adalah mengisi masing-masing tabel yang telah dibuat tadi. Berupa tabel dimensi, tabel proses dan tabel fakta yang ada pada *SQL Server Management* sebelumnya. Langkah awal yang harus dilakukan adalah buka program *SQL Server Business Interlligence Development Studio*. Klik file, pilih *new*, lalu project. Kemudian pilih *integration services project*, ubah name dengan pembuatan tabel dimensi dan fakta. Kemudian *location* untuk lokasi penyimpanan, lalu klik OK. Berikut tampilan seperti gambar 5.33 dibawah ini:



**Gambar 5.33 Project Pembuatan Tabel Dimensi dan Fakta**

Pada gambar 5.33 merupakan tampilan awal pada *SQL Server Business Interlligence Development Studio*. Pada tampilan terdiri dari berbagai pilihan, untuk melakukan analisis data pilih Integration Service Project. Pastikan lokasi penyimpanan data sesuai lalu klik OK setelah memastikan project benar.

Langkah selanjutnya adalah ganti nama *Package* dengan dim\_namafasilitas. Kemudian klik data *flow*, kemudian pada menu *toolbox* tarik OLE DB *Source* dan OLE DB *Destination*. OLE DB *Source* ini berguna sebagai tempat sumber data, sedangkan OLE DB *Destination* berguna sebagai tujuan memindahkan data. Hubungkan keduanya dengan cara klik salah satu tanda panah pada OLE DB *Source* lalu tarik ke OLE DB *Destination*. . Berikut ini dapat dilihat pada gambar 5.34 dibawah ini:

****

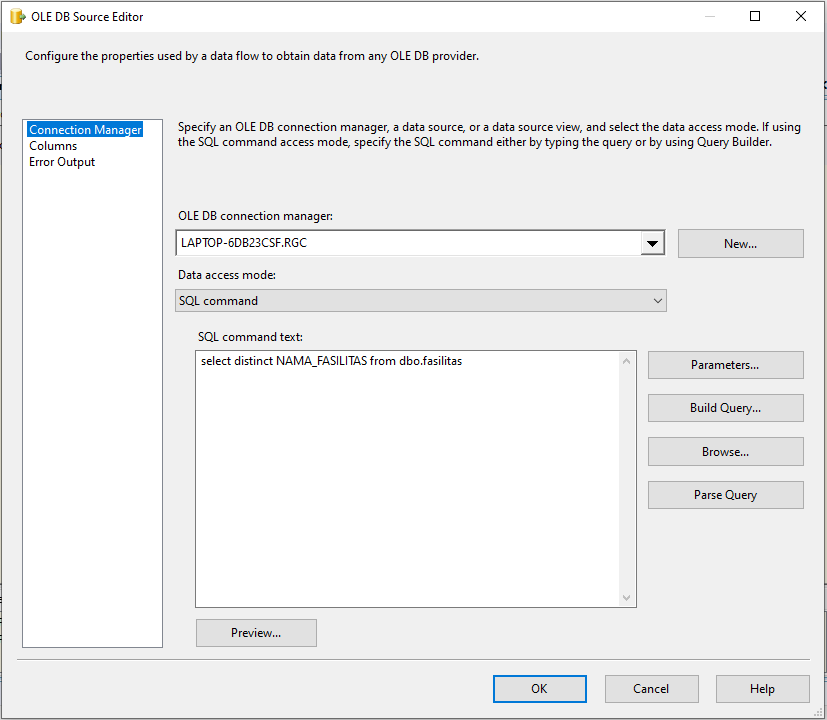
**Gambar 5.34 Menghubungkan *OLE DB Source* dengan *OLE DB Destination***

Pada gambar 5.34 merupakan proses pada tahap data *flow* untuk menghubungkan OLE DB *Source* sebagai tempat penyimpanan *source* dengan OLE DB *Destination* penyimpanan destinasi steelah diproses dengan data *flow.*

Kemudian klik *double* OLE DB *Source* untuk mengedit. Pada OLE DB *connection manager* klik new untuk mencari *database* yang akan digunakan sebagai sumber data, lalu pilih db\_penjualan dan klik OK. Kemudian pada *data access mode* klik tanda panah lalu pilih *SQL* Command. Berikut ini adalah *query* untuk menghilangkan pengulangan nomor part yang ada pada column *NAMA\_FASILITAS*  pada *database* RGC :

select distinct "NAMA\_FASILITAS" from dbo.fasilitas

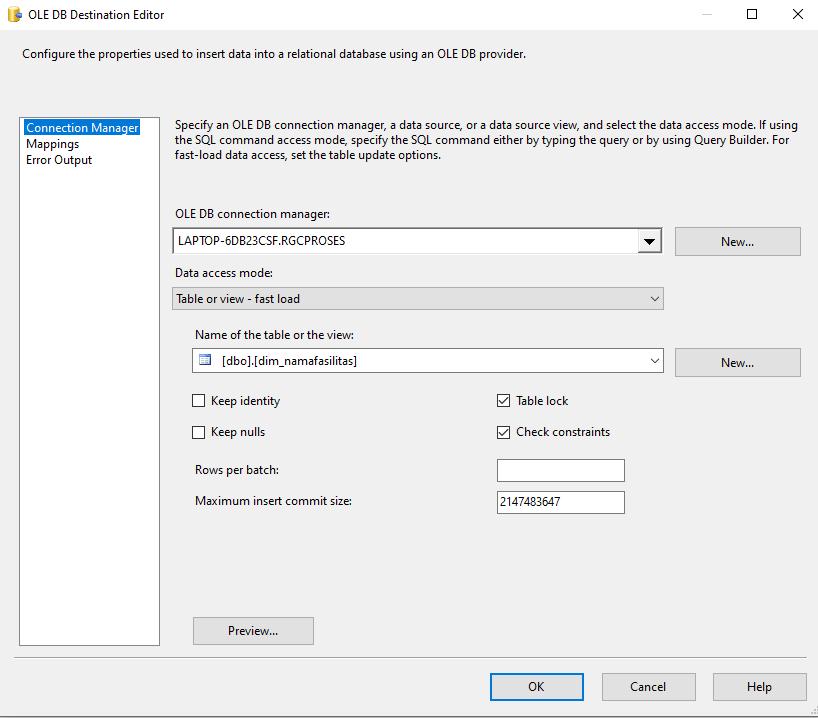
kemudian klik *preview* untuk melihat, lalu klik OK. . Berikut ini dapat dilihat pada gambar 5.35 dibawah ini:



**Gambar 5.35 Mengedit OLE DB Source dim\_namafasilitas**

Pada gambar 5.35 merupakan SQL Command test pada dimensi nama fasilitas. Klik *preview* untuk mengetahui hasil dari SQL Command yang telah dengan benar.

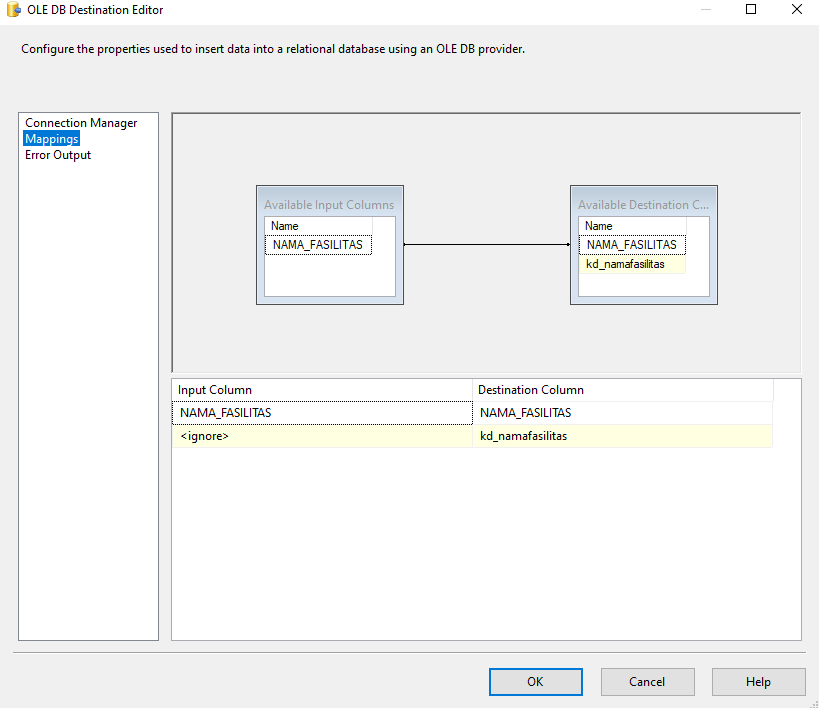
Langkah selanjutnya klik *double* OLE DB *Destination*. Kemudian pada OLE DB *connection manager* klik *new* lalu pilih *database* RGCPROSES dan klik OK. Kemudian pada name of table or the view klik tanda panah lalu pilih dim\_namafasilitas, karena kita ingin memindahkan *NAMA FASILITAS* ke dalam *column NAMA\_FASILITAS* yang ada pada dim\_namafasilitas. Berikut ini dapat dilihat pada gambar 5.36 dibawah ini:



**Gambar 5.36 OLE DB *Destination* dim\_namafasilitas**

Pada gambar 5.36 merupakan tampilan utama setelah klik OLE DB *Destination*. OLE DB *Connection Manager* merupakan koneksi untuk memasukan data kedalam *database* sebagai media penyimpanan. Sesuaikan *name of the table view* dengan data yang telah dilah pada OLE DB *Source*.

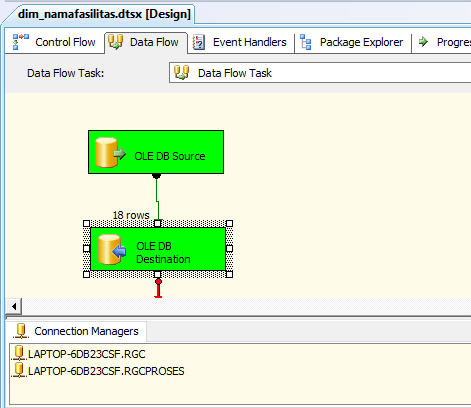
Setelah itu klik *mappings*. Untuk melihat apakah data telah terhubung atau belum. Jika data sudah terhubung klik OK. Berikut ini dapat dilihat pada gambar 5.37 dibawah ini:

****

**Gambar 5.37 *Mapping* dim\_namafasilitas**

Pada gambar 5.37 merupakan tampilan *mapping* pada OLE DB *Destination*. Perhatikan tanda penghubung antara tabel dari input coloums dengan *destination* *coloums* terhubung dengan benar. Klik OK jika telah selesai.

Kemudian klik tombol *start debugging*. Jika warna hijau berarti data berhasil dipindahkan. . Berikut ini dapat dilihat pada gambar 5.38 dibawah ini:



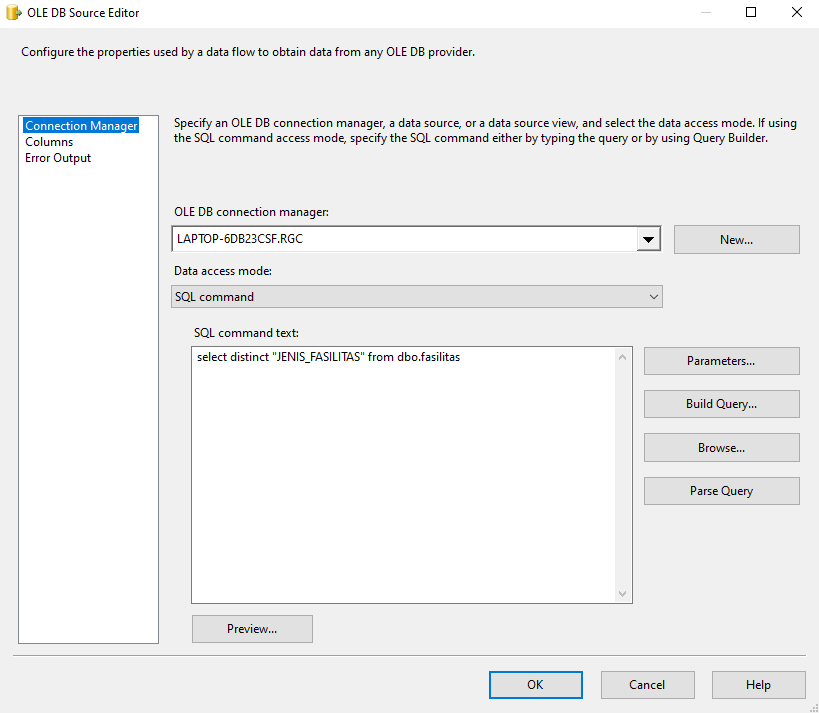
**Gambar 5.38 Berhasil Memindahkan dim\_namafasilitas**

Pada gambar 5.38 merupakan tampilan *debug* yang telah sukses dan data telah masuk kedalam *database* RGCPROSES.

Langkah selanjutnya klik kanan pada SSIS *Package*, pilih *new* SSIS *Package*. Kemudian ganti nama *Package* dengan dim\_jenisfasilitas. Lalu klik *data flow*. Kemudian pilih *toolbox*, tarik OLE DB *Source* dan OLE DB *Destination* lalu hubungkan dengan cara menarik tanda panah dari OLE DB *Source* ke OLE DB *Destination*. Kemudian *double* klik OLE DB *Source* pada OLE DB *connection manager* klik new, pilih *database* RGC, lalu OK. Kemudian pada *data access mode* klik tanda panah lalu pilih *SQL* Command. Ini adalah *query* untuk menghilangkan pengulangan nama satuan yang ada pada column satuan pada *database* RGC sebagai berikut:

select distinct "JENIS\_FASILITAS" from dbo.fasilitas

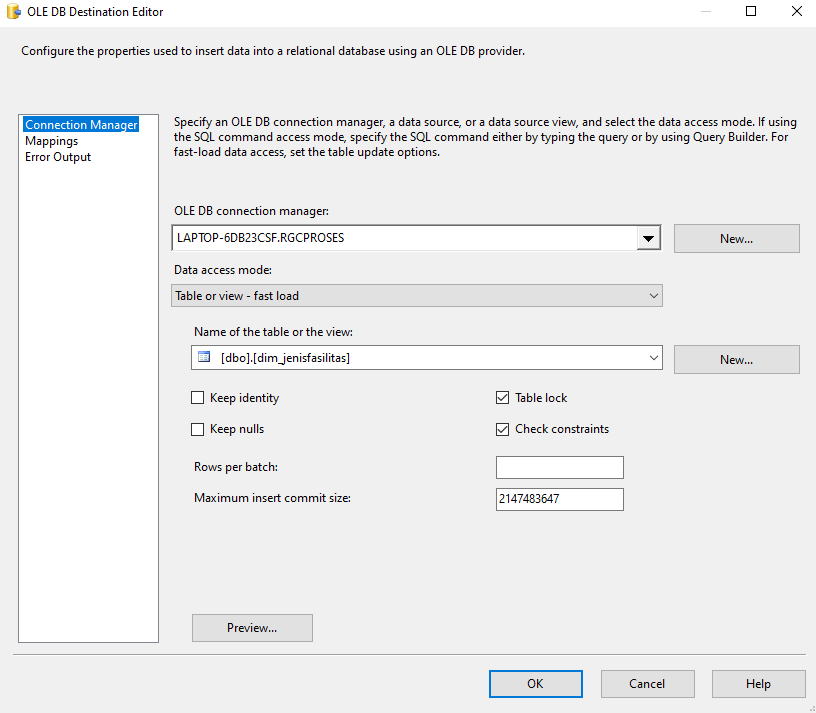
kemudian klik *preview* untuk melihat, lalu klik OK. . Berikut ini dapat dilihat pada gambar 5.39 dibawah ini:

****

**Gambar 5.39 OLE DB Source dim\_jenisfasilitas**

Pada gambar 5.39 merupakan SQL Command test pada dimensi jenis fasilitas. Klik *preview* untuk mengetahui hasil dari SQL Command yang telah dengan benar.

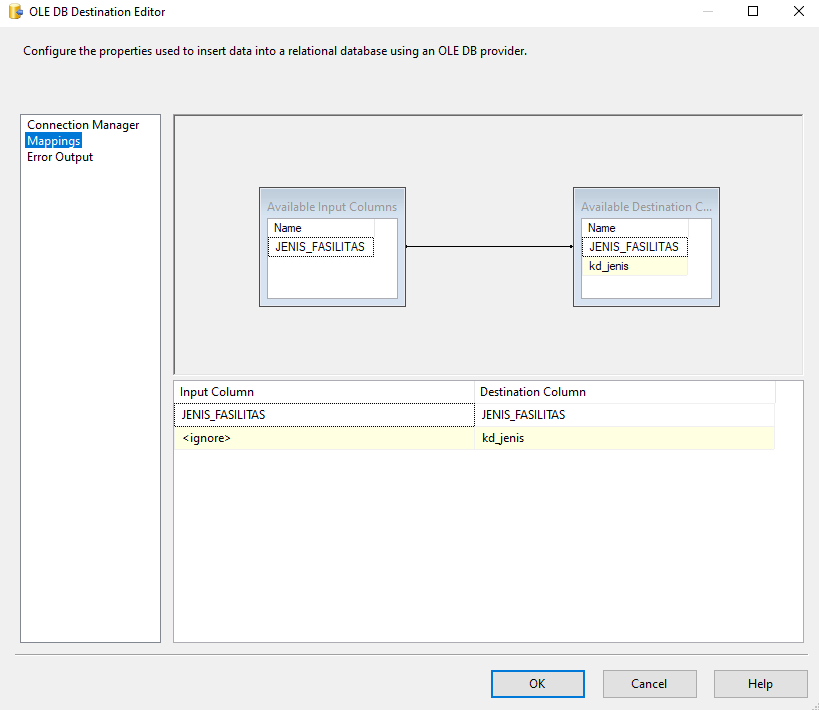
Langkah selanjutnya klik *double* OLE DB *Destination*. Kemudian pada OLE DB *connection manager* klik *new* lalu pilih *database* RGCPROSES dan klik OK. Kemudian pada *name of table or the view* klik tanda panah lalu pilih dim\_namafasilitas, karena kita ingin memindahkan *JENIS FASILITAS* ke dalam *column JENIS\_FASILITAS* yang ada pada dim\_jenisfasilitas. . Berikut ini dapat dilihat pada gambar 5.40 dibawah ini:



**Gambar 5.40 OLE DB Destination dim\_jenisfasilitas**

Pada gambar 5.40 merupakan tampilan utama setelah klik OLE DB *Destination*. OLE DB *Connection Manager* merupakan koneksi untuk memasukan data kedalam *database* sebagai media penyimpanan. Sesuaikan *name of the table view* dengan data yang telah diolah pada OLE DB *Source*.

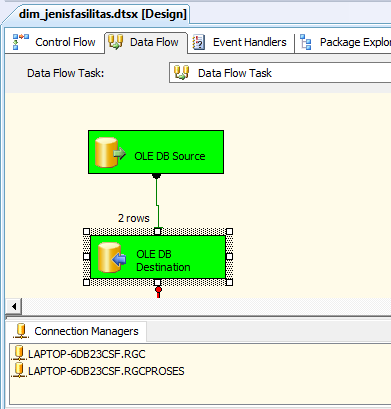
Kemudian klik *mapping*. Jika sudah terhubung, klik OK. Berikut ini dapat dilihat pada gambar 5.41 dibawah ini:

****

**Gambar 5.41 *Mapping* dim\_jenisfasilitas**

Pada gambar 5.41 merupakan tampilan *mapping* pada OLE DB *Destination*. Perhatikan tanda penghubung antara tabel dari input coloums dengan *destination coloums* terhubung dengan benar. Klik OK jika telah selesai

Kemudian klik tombol *start debugging*. Jika warna hijau berarti data berhasil dipindahkan. . Berikut ini dapat dilihat pada gambar 5.42 dibawah ini:



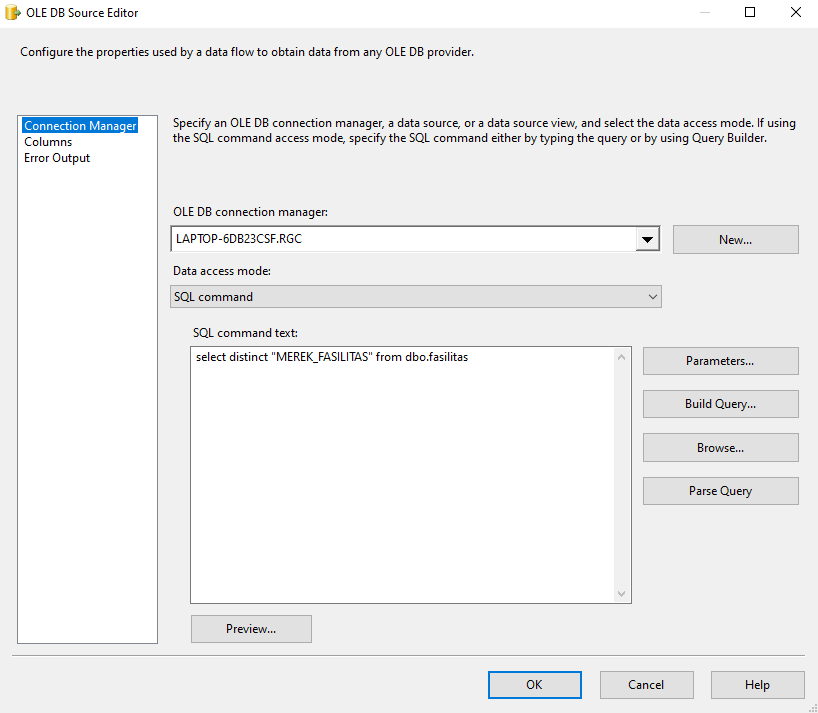
**Gambar 5.42 Berhasil Memindahkan dim\_jenisfasilitas**

Pada gambar 5.42 merupakan tampilan *debug* yang telah sukses dan data telah masuk kedalam database RGCPROSES.

Langkah selanjutnya klik kanan pada SSIS *Package*, pilih *new* SSIS *Package*. Kemudian ganti nama *Package* dengan dim\_merekfasilitas. Lalu klik *data flow*. Kemudian pilih *toolbox*, tarik OLE DB *Source* dan OLE DB *Destination* lalu hubungkan dengan cara menarik tanda panah dari OLE DB *Source* ke OLE DB *Destination*. Kemudian *double* klik OLE DB *Source* pada OLE DB *connection manager* klik new, pilih *database* RGC, lalu OK. Kemudian pada *data access mode* klik tanda panah lalu pilih *SQL* Command. Berikut ini adalah *query* yang ada pada column satuan pada *database* RGC sebagai berikut:

select distinct "MEREK\_FASILITAS" from dbo.fasilitas

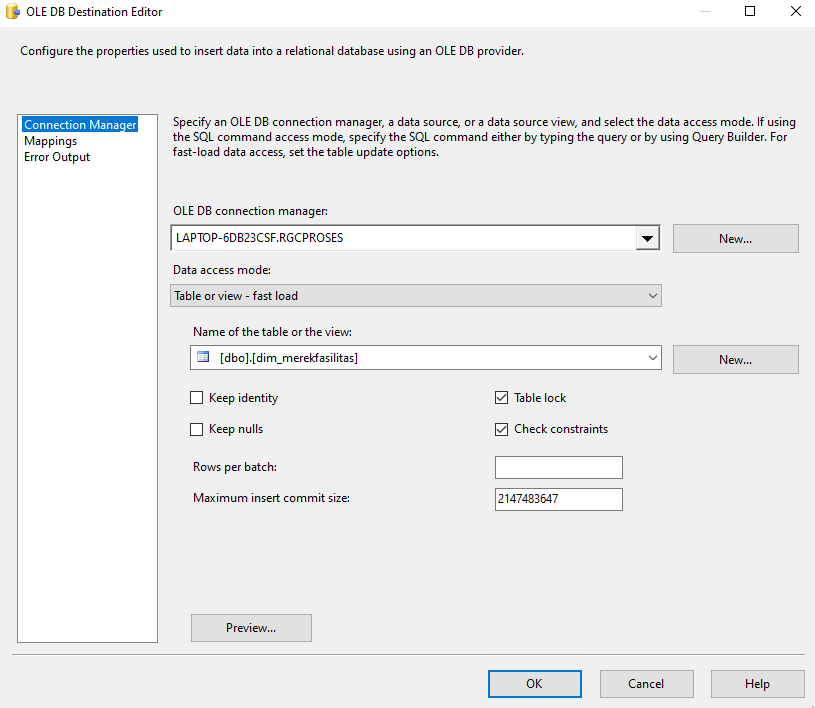
kemudian klik *preview* untuk melihat, lalu klik OK. . Berikut ini dapat dilihat pada gambar 5.43 dibawah ini:



**Gambar 5.43 Mengedit OLE DB Source dim\_merekfasilitas**

Pada gambar 5.43 merupakan SQL Command test pada dimensi merek fasilitas. Klik *preview* untuk mengetahui hasil dari SQL Command yang telah dengan benar.

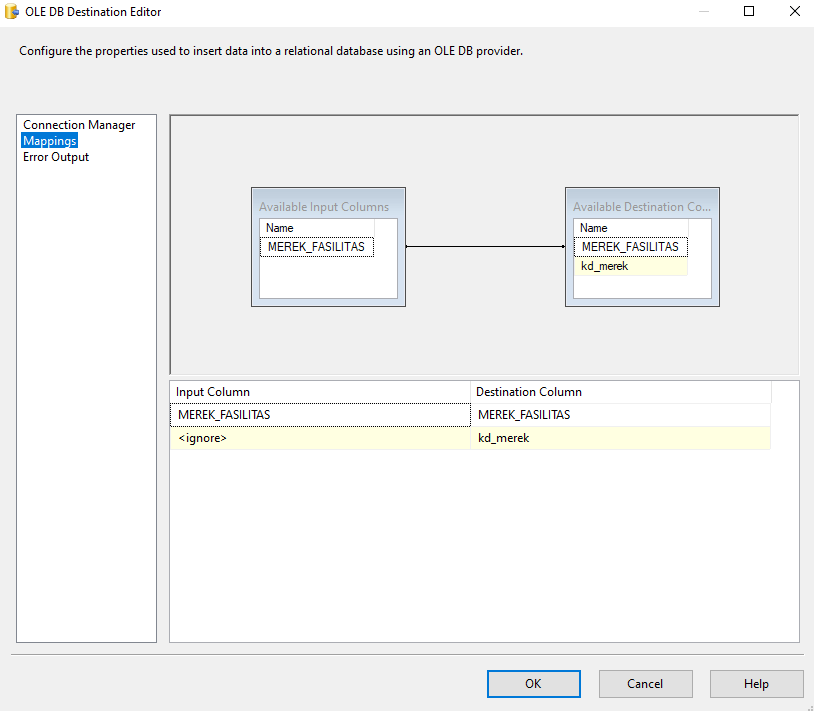
Langkah selanjutnya klik *double* OLE DB *Destination*. Kemudian pada OLE DB *connection manager* klik *new* lalu pilih *database* RGCPROSES dan klik OK. Kemudian pada name of table or the view klik tanda panah lalu pilih dim\_namafasilitas, karena kita ingin memindahkan *MEREK FASILITAS* ke dalam *column MEREK\_FASILITAS* yang ada pada dim\_merekfasilitas. Berikut ini dapat dilihat pada gambar 5.44 dibawah ini:



**Gambar 5.44 OLE DB Destination dim\_merekfasilitas**

Pada gambar 5.44 merupakan tampilan utama setelah klik OLE DB *Destination*. OLE DB *Connection Manager* merupakan koneksi untuk memasukan data kedalam *database* sebagai media penyimpanan. Sesuaikan *name of the table view* dengan data yang telah diolah pada OLE DB *Source*. Pilih [dbo].[dim\_merekfasilitas] sesuai dengan dimensi yang menjadi penghubung.

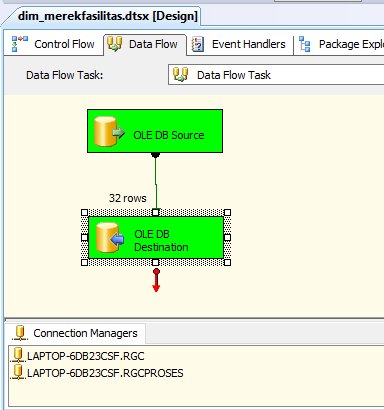
Kemudian klik *mapping*. Jika sudah terhubung, klik OK. Berikut ini dapat dilihat pada gambar 5.45 dibawah ini:

****

**Gambar 5.45 *Mapping* dim\_merekfasilitas**

Pada gambar 5.45 merupakan tampilan *mapping* pada OLE DB *Destination*. Perhatikan tanda penghubung antara tabel dari input coloums dengan *destination coloums* terhubung dengan benar. Klik OK jika telah selesai.

Kemudian klik tombol *start debugging*. Jika warna hijau berarti data berhasil dipindahkan. Berikut ini dapat dilihat pada gambar 5.46 dibawah ini:



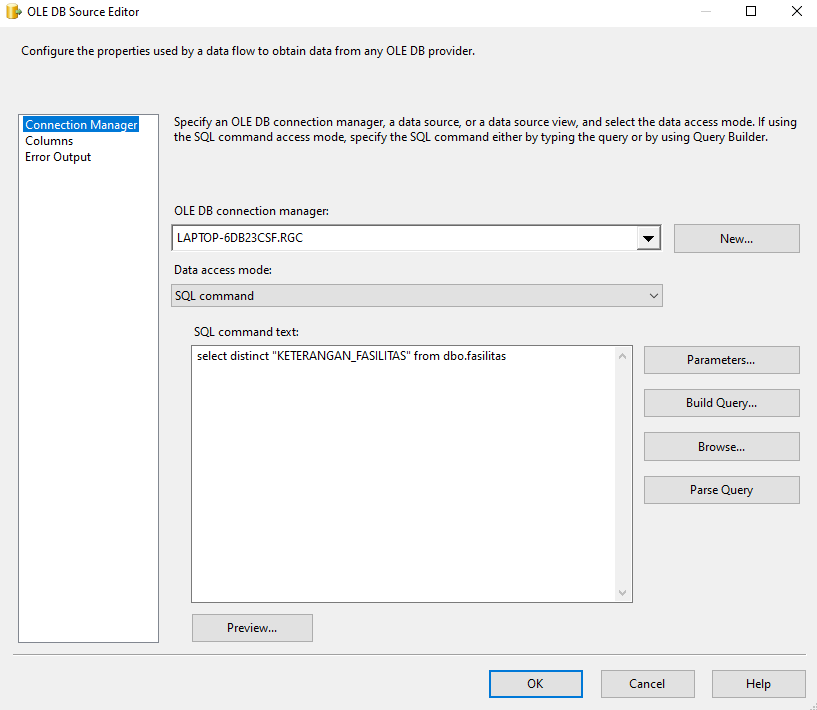
**Gambar 5.46 Berhasil Memindahkan dim\_merekfasilitas**

Pada gambar 5.46 merupakan tampilan *debug* yang telah sukses dan data telah masuk kedalam database RGCPROSES.

Langkah selanjutnya klik kanan pada SSIS *Package*, pilih *new* SSIS *Package*. Kemudian ganti nama *Package* dengan dim\_keteranganfasilitas. Lalu klik *data flow*. Kemudian pilih *toolbox*, tarik OLE DB *Source* dan OLE DB *Destination* lalu hubungkan dengan cara menarik tanda panah dari OLE DB *Source* ke OLE DB *Destination*. Kemudian *double* klik OLE DB *Source* pada OLE DB *connection manager* klik new, pilih *database* RGC, lalu OK. Kemudian pada *data access mode* klik tanda panah lalu pilih *SQL* Command. Berikut ini adalah *query* untuk menghilangkan pengulangan nama satuan yang ada pada column satuan pada *database* RGC sebagai berikut:

select distinct "KETERANGAN\_FASILITAS" from dbo.fasilitas

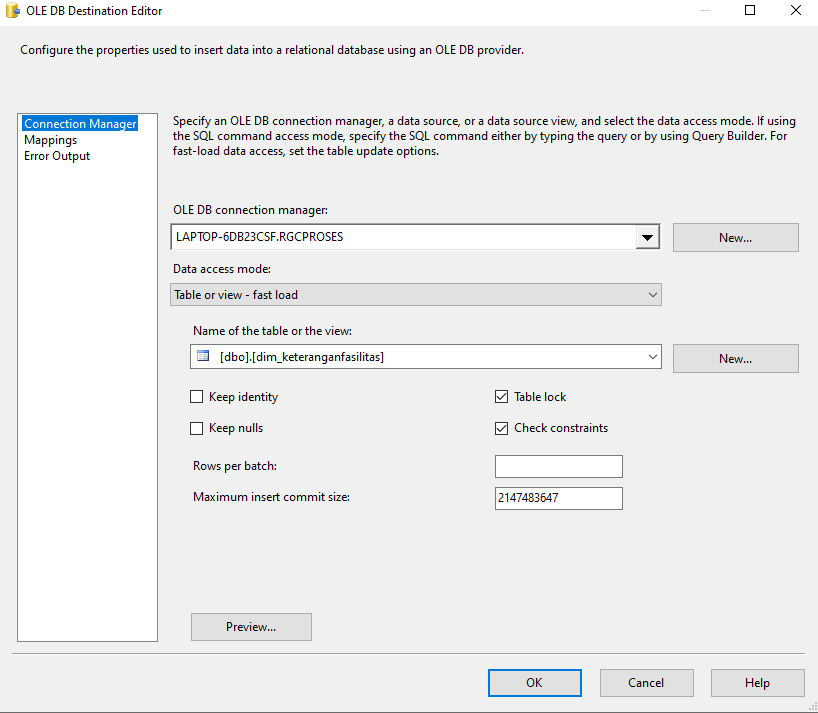
kemudian klik *preview* untuk melihat, lalu klik OK. Berikut ini dapat dilihat pada gambar 5.47 dibawah ini:

****

**Gambar 5.47 Mengedit OLE DB Source dim\_keteranganfasilitas**

Pada gambar 5.47 merupakan SQL Command test pada dimensi keterangan fasilitas. Klik *preview* untuk mengetahui hasil dari SQL Command yang telah dengan benar.

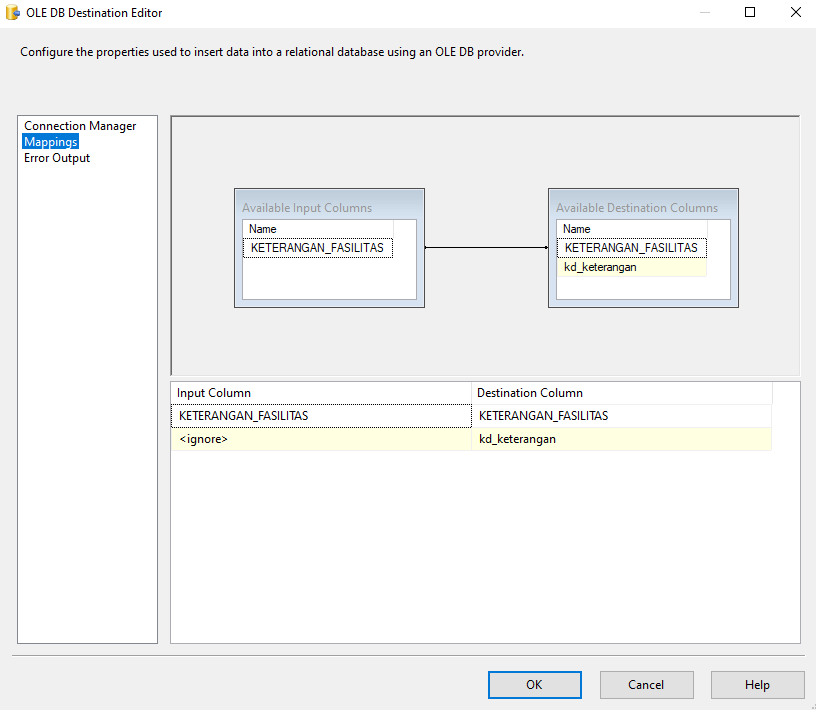
Langkah selanjutnya klik *double* OLE DB *Destination*. Kemudian pada OLE DB *connection manager* klik *new* lalu pilih *database* RGCPROSES dan klik OK. Kemudian pada name of table or the view klik tanda panah lalu pilih dim\_namafasilitas, karena kita ingin memindahkan *KETERANGAN FASILITAS* ke dalam *column KETERANGAN\_FASILITAS* yang ada pada dim\_ketfasilitas. Berikut ini dapat dilihat pada gambar 5.48 dibawah ini:



**Gambar 5.48 OLE DB Destination dim\_keteranganfasilitas**

Pada gambar 5.48 merupakan tampilan utama setelah klik OLE DB *Destination*. OLE DB *Connection Manager* merupakan koneksi untuk memasukan data kedalam *database* sebagai media penyimpanan. Sesuaikan *name of the table view* dengan data yang telah diolah pada OLE DB *Source*. Pilih [dbo].[dim\_keterangankfasilitas] sesuai dengan dimensi yang menjadi penghubung.

Kemudian klik *mapping*. Jika sudah terhubung, klik OK. Berikut ini dapat dilihat pada gambar 5.49 dibawah ini:

****

**Gambar 5.49 *Mapping* dim\_keteranganfasilitas**

Pada gambar 5.49 merupakan tampilan *mapping* pada OLE DB *Destination*. Perhatikan tanda penghubung antara tabel dari input coloums dengan *destination coloums* terhubung dengan benar. Klik OK jika telah selesai.

Setelah itu klik tombol *start debugging*. Jika warna hijau berarti data berhasil dipindahkan. Berikut ini dapat dilihat pada gambar 5.50 dibawah ini:

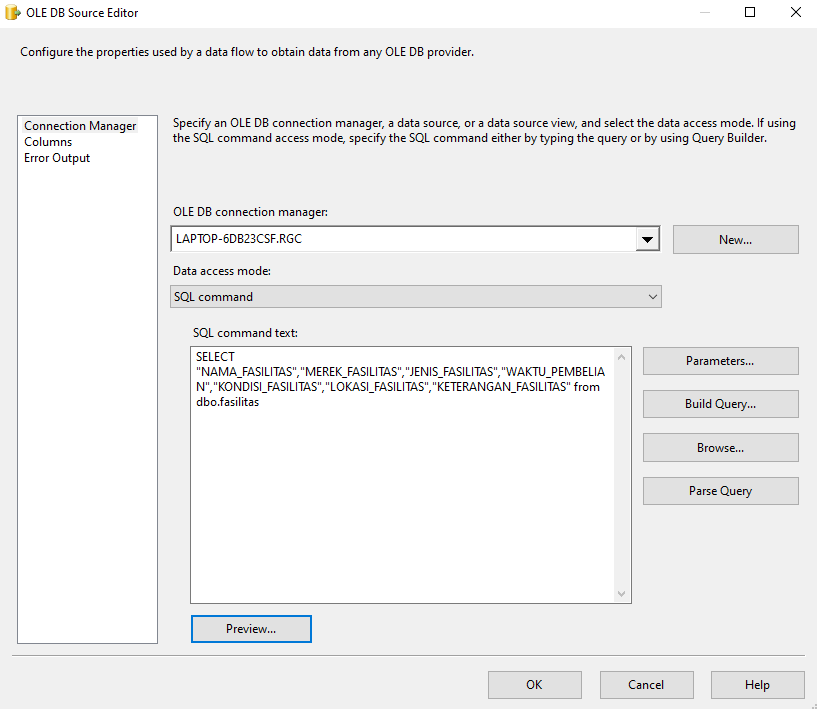


**Gambar 5.50 Berhasil Memindahkan dim\_keteranganfasilitas**

Pada gambar 5.50 merupakan tampilan *debug* yang telah sukses dan data telah masuk kedalam database RGCPROSES.

Lakukan langkah-langkah diatas untuk seluruh dimensi kondisi fasilitas, dimensi instansi, dimensi tenaga ahli, dimensi tingkat pendidikan, dimensi jurusan, dimensi universitas, dimensi status, dimensi sub bidang, dimensi pemberi jasa, dimensi daerah, dimensi kabupaten , dimensi provinsi dan dimensi ibu kota.

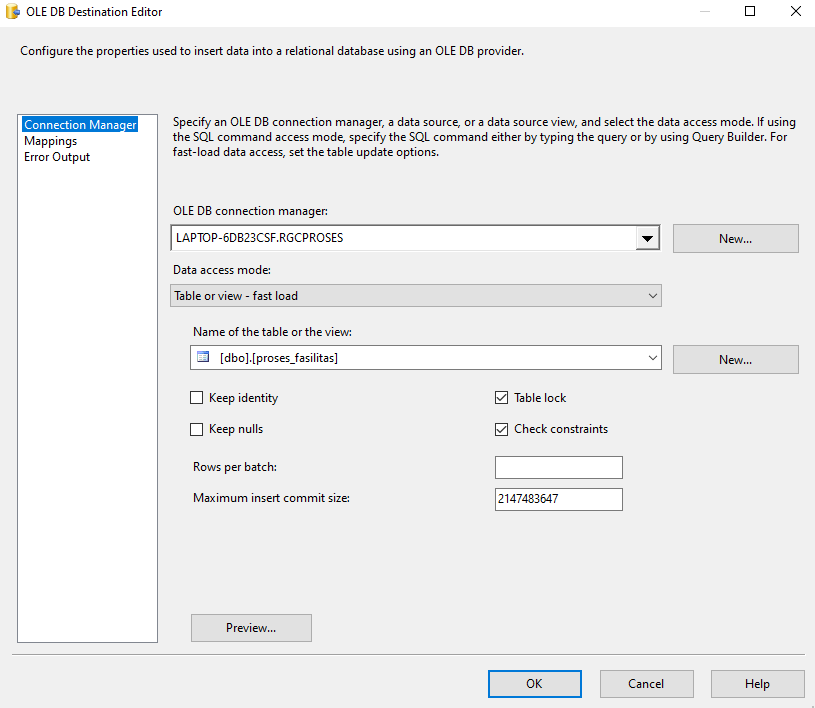
Langkah selanjutnya adalah mengambil data dari *database* RGC untuk dipindahkan ke table proses\_fasilitas. Caranya adalah klik *new SSIS Package*, kemudian ganti namanya dengan proses\_fasilitas. Kemudian klik *data flow*, pada *toolbox* tarik OLE DB *Source* dan OLE DB *Destination*, lalu tarik tanda panah dari OLE DB *Source* kearah OLE DB *Destination* untuk menghubungkannya. Klik *double* OLE DB *Source* lalu pada OLE DB *connection manager* pilih *new*, lalu pilih *database* RGC kemudian klik OK. Pada *Data type* pilih *SQL* Command. Kemudian ketikan *query* berikut pada kotak *query* klik *preview*, lalu OK. Berikut ini dapat dilihat pada gambar 5.51 dibawah ini:

****

**Gambar 5.51 OLE DB *Source* proses\_fasilitas**

Pada gambar 5.51 merupakan SQL Command untuk memanggil data pada database RGC.

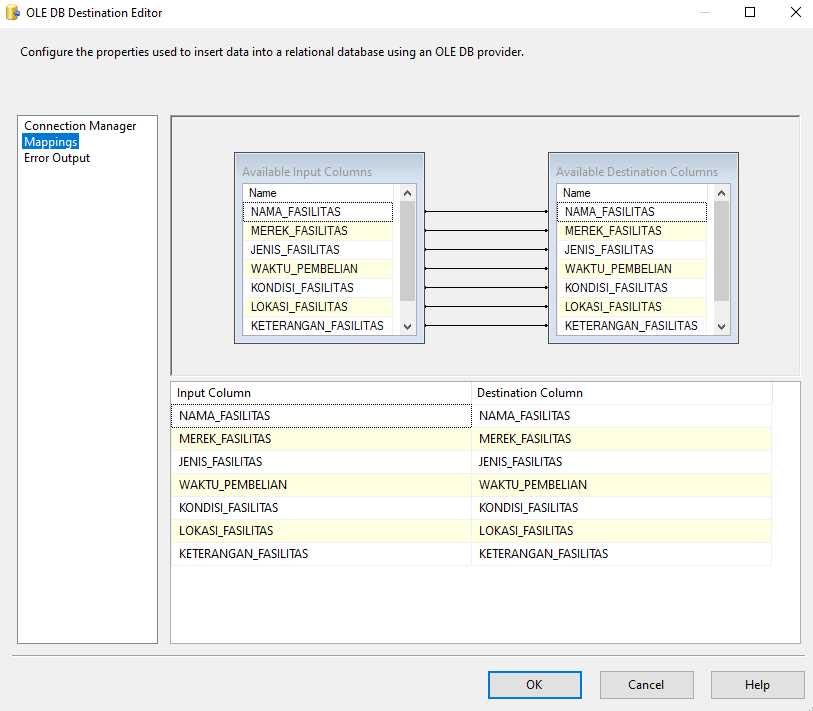
Lalu klik double OLE DB *Destination*. Kemudian klik *new*, pilih *database* RGCPROSES, lalu OK. Berikut ini dapat dilihat pada gambar 5.52 dibawah ini:

****

**Gambar 5.52 OLE DB Destination proses\_fasilitas**

Pada gambar 5.52 merupakan tampilan utama setelah klik OLE DB *Destination*. OLE DB *Connection Manager* merupakan koneksi untuk memasukan data kedalam *database* sebagai media penyimpanan. Sesuaikan *name of the table view* dengan data yang telah diolah pada OLE DB *Source*.

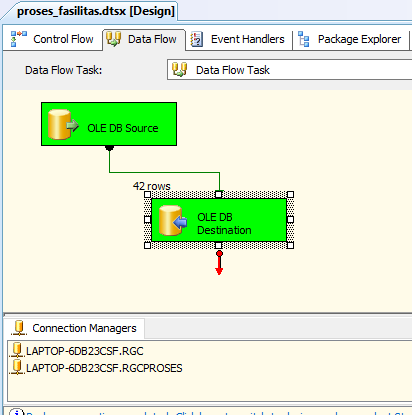
Kemudian klik *Mapping* untuk melihat apakah data itu telah terhubung atau belum. Jika sudah klik OK. Berikut ini dapat dilihat pada gambar 5.53 dibawah ini:

****

**Gambar 5.53 *Mapping* proses\_fasilitas**

Pada gambar 5.53 merupakan tampilan *mapping* pada OLE DB *Destination*. Perhatikan tanda penghubung antara tabel dari input coloums dengan *destination coloums* terhubung dengan benar. Klik OK jika telah selesai.

Setelah melakukan *mapping* maka klik *start debugging* untuk menjalankan proses pemindahan. Jika berhasil maka akan berwarna hijau. Berikut ini dapat dilihat pada gambar 5.54 dibawah ini:



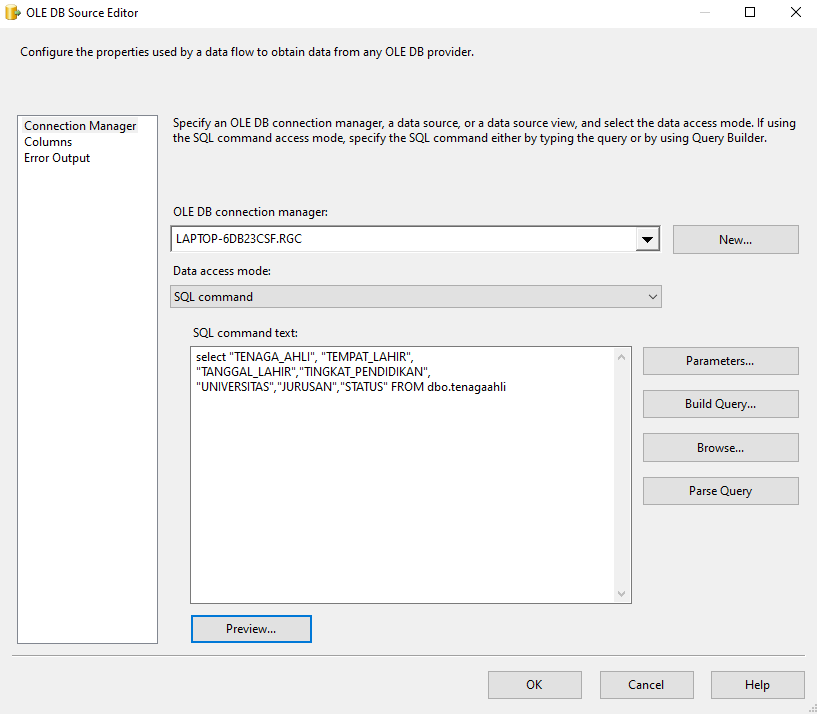
**Gambar 5.54 Berhasil Memindahkan Data ke proses\_fasilitas**

Pada gambar 5.54 merupakan tampilan *debug* yang telah sukses dan data telah masuk kedalam database RGCPROSES.

Langkah selanjutnya adalah mengambil data dari *database* RGC untuk dipindahkan ke table proses\_tenagaahli. Caranya adalah klik *new SSIS Package*, kemudian ganti namanya dengan proses\_tenagaahli. Kemudian klik *data flow*, pada *toolbox* tarik OLE DB *Source* dan OLE DB *Destination*, lalu tarik tanda panah dari OLE DB *Source* kearah OLE DB *Destination* untuk menghubungkannya. Klik *double* OLE DB *Source* lalu pada OLE DB *connection manager* pilih *new*, lalu pilih *database* RGC kemudian klik OK.

Pada *Data type* pilih *SQL* Command. Kemudian ketikan *query* berikut pada kotak *query* :

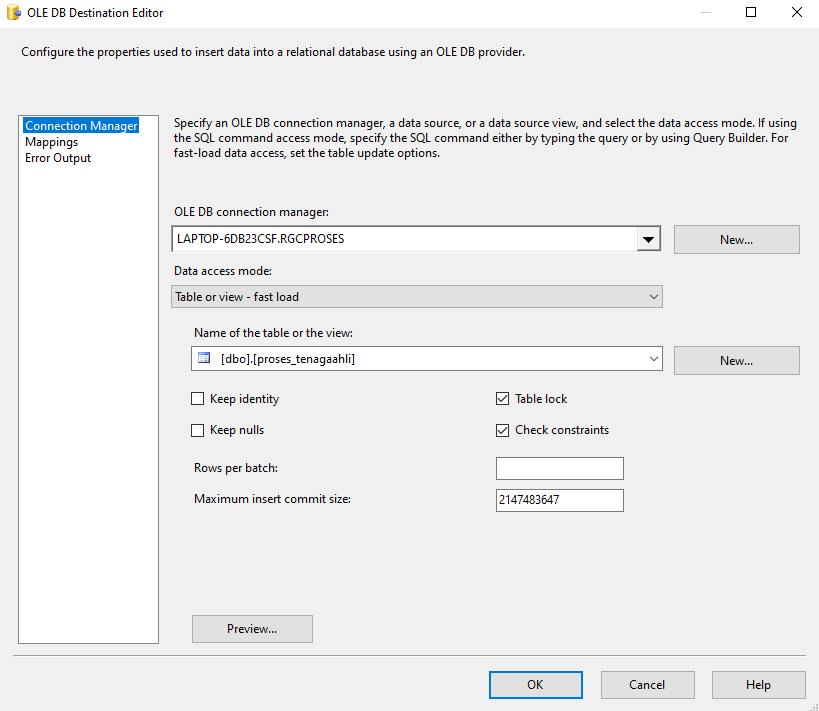
Select “TENAGA\_AHLI”, ”TEMPAT\_LAHIR”, ”TANGGAL\_LAHIR”, “TINGKAT\_PENDIDIKAN”, ”JURUSAN”, “UNIVERISTAS”, ”STATUS” from dbo.tenagaahli klik *preview*, lalu OK. Berikut ini dapat dilihat pada gambar 5.55 dibawah ini:

****

**Gambar 5.55 OLE DB *Source* proses\_tenagaahli**

Pada gambar 5.55 merupakan SQL Command test pada tabel proses tenaga ahli. Klik *preview* untuk mengetahui hasil dari SQL Command yang telah dengan benar.

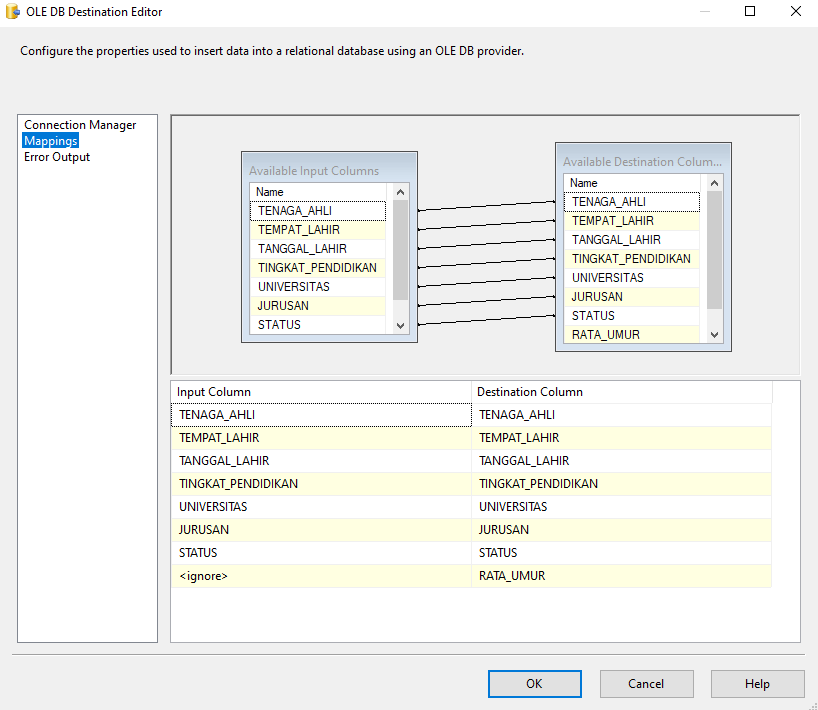
Langkah selanjutnya adalah klik double OLE DB *Destination*. Kemudian klik *new*, pilih *database* RGCPROSES, lalu OK. Kemudian pada *name of table or view* pilih [dbo].[proses\_tenagaahli], lalu klik *preview*. Berikut ini dapat dilihat pada gambar 5.56 dibawah ini:

****

**Gambar 5.56 OLE DB Destination proses\_tenagaahli**

Pada gambar 5.56 merupakan tampilan utama setelah klik OLE DB *Destination*. OLE DB *Connection Manager* merupakan koneksi untuk memasukan data kedalam *database* sebagai media penyimpanan. Sesuaikan *name of the table view* dengan data yang telah diolah pada OLE DB *Source*.

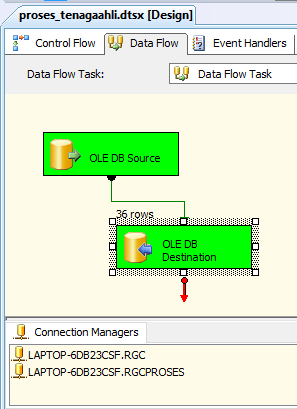
Kemudian klik *Mapping* untuk melihat apakah data itu telah terhubung atau belum. Jika sudah klik OK. Berikut ini dapat dilihat pada gambar 5.57 dibawah ini:

****

**Gambar 5.57 *Mapping* proses\_tenagaahli**

Pada gambar 5.57 merupakan tampilan *mapping* pada OLE DB *Destination*. Perhatikan tanda penghubung antara tabel dari input coloums dengan *destination coloums* terhubung dengan benar. Klik OK jika telah selesai.

Setelah selesai *mapping* maka klik start debugging untuk menjalankan proses pemindahan. Jika berhasil maka akan berwarna hijau seperti gambar 5.58 berikut:

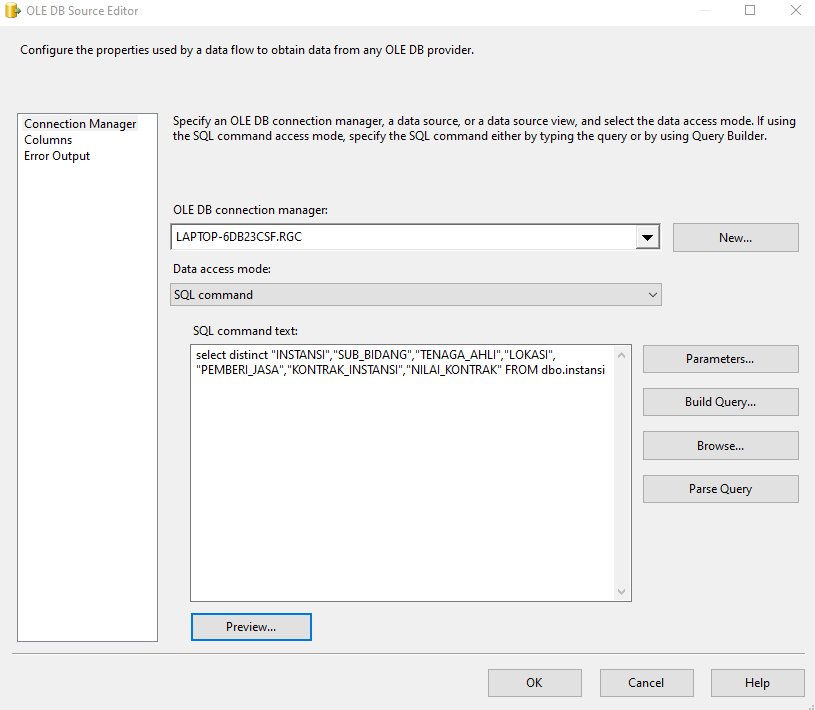


**Gambar 5.58 Berhasil Memindahkan Data ke proses\_tenagaahli**

Pada gambar 5.58 merupakan tampilan *debug* yang telah sukses dan data telah masuk kedalam database RGCPROSES.

Langkah selanjutnya adalah mengambil data dari *database* RGC untuk dipindahkan ke table proses\_instansi. Caranya adalah klik *new SSIS Package*, kemudian ganti namanya dengan proses\_instansi. Kemudian klik *data flow*, pada *toolbox* tarik OLE DB *Source* dan OLE DB *Destination*, lalu tarik tanda panah dari OLE DB *Source* kearah OLE DB *Destination* untuk menghubungkannya. Klik *double* OLE DB *Source* lalu pada OLE DB *connection manager* pilih *new*, lalu pilih *database* RGC kemudian klik OK. Pada *Data type* pilih *SQL* Command. Kemudian ketikan *query* berikut pada kotak *query* :

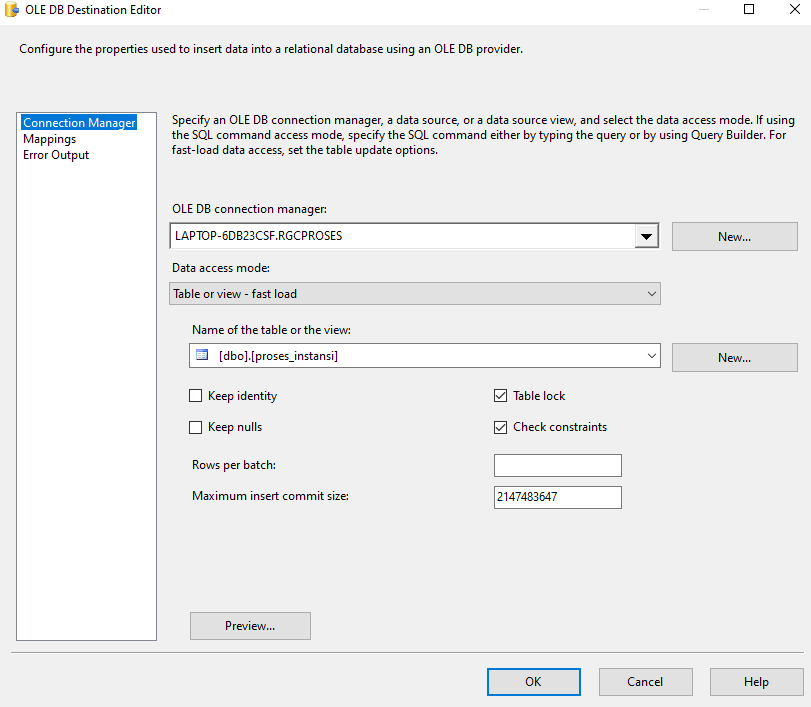
Select “INSTANSI”, ”SUB\_BIDANG”, ”TENAGA\_AHLI”, “LOKASI”, ”PEMBERI\_JASA”, “KONTRAK\_INSTANSI”, ”NILAI\_KONTRAK” from dbo.instansi klik *preview*, lalu OK. Berikut ini dapat dilihat pada gambar 5.59 dibawah ini:

****

**Gambar 5.59 OLE DB *Source* proses\_instansi**

Pada gambar 5.59 merupakan SQL Command test pada tabel proses instansi. Klik *preview* untuk mengetahui hasil dari SQL Command yang telah dengan benar.

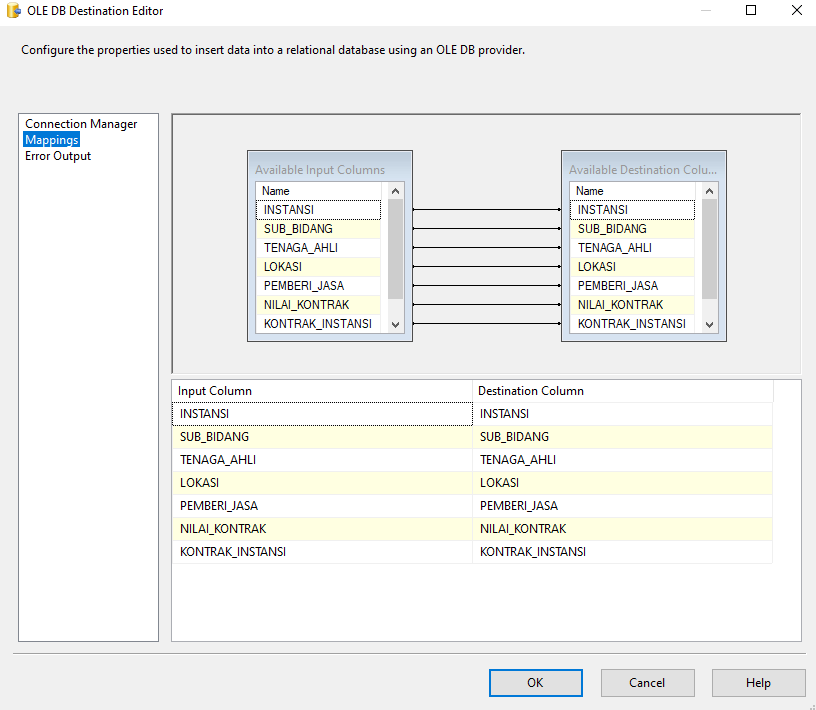
Langkah selanjutnya adalah klik double OLE DB *Destination*. Kemudian klik *new*, pilih *database* RGCPROSES, lalu OK. Kemudian pada *name of table or view* pilih [dbo].[proses\_instansi], lalu klik *preview*. Berikut ini dapat dilihat pada gambar 5.60 dibawah ini:

****

**Gambar 5.60 OLE DB Destination proses\_instansi**

Pada gambar 5.60 merupakan tampilan utama setelah klik OLE DB *Destination*. OLE DB *Connection Manager* merupakan koneksi untuk memasukan data kedalam *database* sebagai media penyimpanan. Sesuaikan *name of the table view* dengan data yang telah diolah pada OLE DB *Source*.

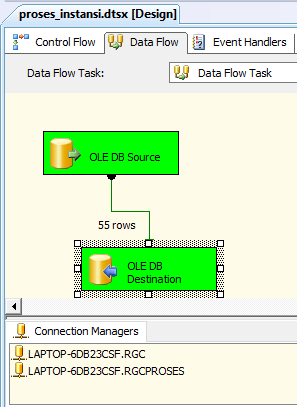
Kemudian klik *Mapping* untuk melihat apakah data itu telah terhubung atau belum. Jika sudah klik OK. Berikut ini dapat dilihat pada gambar 5.61 dibawah ini:

****

**Gambar 5.61 *Mapping* proses\_instansi**

Pada gambar 5.61 merupakan tampilan *mapping* pada OLE DB *Destination*. Perhatikan tanda penghubung antara tabel dari input coloums dengan *destination coloums* terhubung dengan benar. Klik OK jika telah selesai.

Setelah selesai *mapping* maka klik *start debugging* untuk menjalankan proses pemindahan. Jika berhasil maka akan berwarna hijau seperti gambar 5.62 berikut:



**Gambar 5.62 Berhasil Memindahkan Data ke proses\_instansi**

Pada gambar 5.62 merupakan tampilan *debug* yang telah sukses dan data telah masuk kedalam database RGCPROSES.

Langkah selanjutnya adalah membuat fakta. Caranya adalah klik *new SSIS Package*, kemudian ganti namanya dengan fakta fasilitas. Kemudian klik *data flow*, pada *toolbox* tarik OLE DB *Source* dan OLE DB *Destination*, lalu tarik tanda panah dari OLE DB *Source* kearah OLE DB *Destination* untuk menghubungkannya. Klik *double* OLE DB *Source* lalu pada OLE DB *connection manager* pilih *new*, lalu pilih db\_proses kemudian klik OK. Pada *Data type* pilih *SQL* Command. Kemudian ketikan *query inner join* berikut pada kotak *query*:

***SELECT DISTINCT***

*dim\_namafasilitas.kd\_namafasilitas,*

*dim\_merekfasilitas.kd\_merek,*

*dim\_jenisfasilitas.kd\_jenis,*

*dim\_kondisifasilitas.kd\_kondisi,*

*dim\_daerah.kd\_daerah,*

*dim\_keteranganfasilitas.kd\_keterangan,*

*dim\_waktu.kd\_waktu*

***FROM proses\_fasilitas***

*INNER JOIN dim\_namafasilitas*

*ON proses\_fasilitas.NAMA\_FASILITAS=dim\_namafasilitas.NAMA\_FASILITAS*

*INNER JOIN dim\_merekfasilitas*

*ON proses\_fasilitas.MEREK\_FASILITAS=dim\_merekfasilitas.MEREK\_FASILITAS*

*INNER JOIN dim\_jenisfasilitas*

*ON proses\_fasilitas.JENIS\_FASILITAS=dim\_jenisfasilitas.JENIS\_FASILITAS*

*INNER JOIN dim\_kondisifasilitas*

*ON proses\_fasilitas.KONDISI\_FASILITAS=dim\_kondisifasilitas.KONDISI\_FASILITAS*

*INNER JOIN dim\_waktu*

*ON proses\_fasilitas.WAKTU\_PEMBELIAN=dim\_waktu.WAKTU*

*INNER JOIN dim\_keteranganfasilitas*

*ON proses\_fasilitas.KETERANGAN\_FASILITAS=dim\_keteranganfasilitas.KETERANGAN\_FASILITAS*

*INNER JOIN dim\_daerah*

*ON proses\_fasilitas.LOKASI\_FASILITAS=dim\_daerah.DAERAH*

***GROUP BY***

*dim\_namafasilitas.kd\_namafasilitas,*

*dim\_merekfasilitas.kd\_merek,*

*dim\_jenisfasilitas.kd\_jenis,*

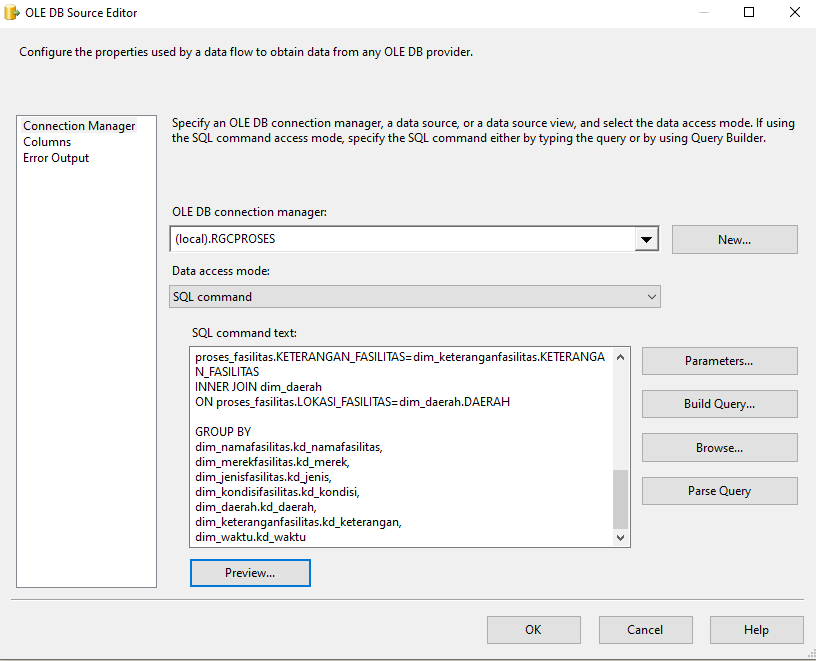
*dim\_kondisifasilitas.kd\_kondisi,*

*dim\_daerah.kd\_daerah,*

*dim\_keteranganfasilitas.kd\_keterangan,*

*dim\_waktu.kd\_waktu*

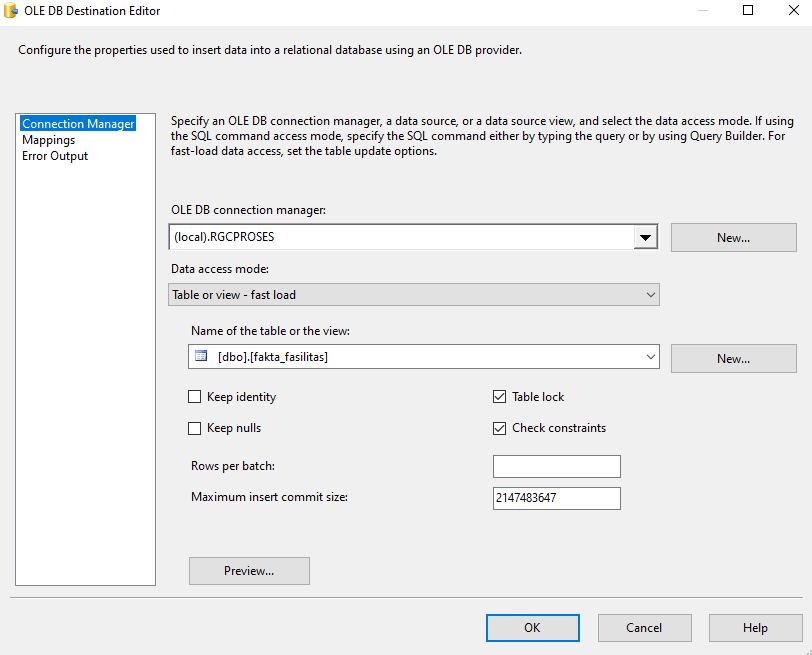
Kemudian klik *preview*, lalu OK. Berikut ini dapat dilihat pada gambar 5.63 dibawah ini:

****

**Gambar 5.63 OLE DB *Source* Fakta Fasilitas**

Pada gambar 5.63 merupakan SQL Command test pada fakta fasilitas. Klik *preview* untuk mengetahui hasil dari SQL Command yang telah dengan benar.

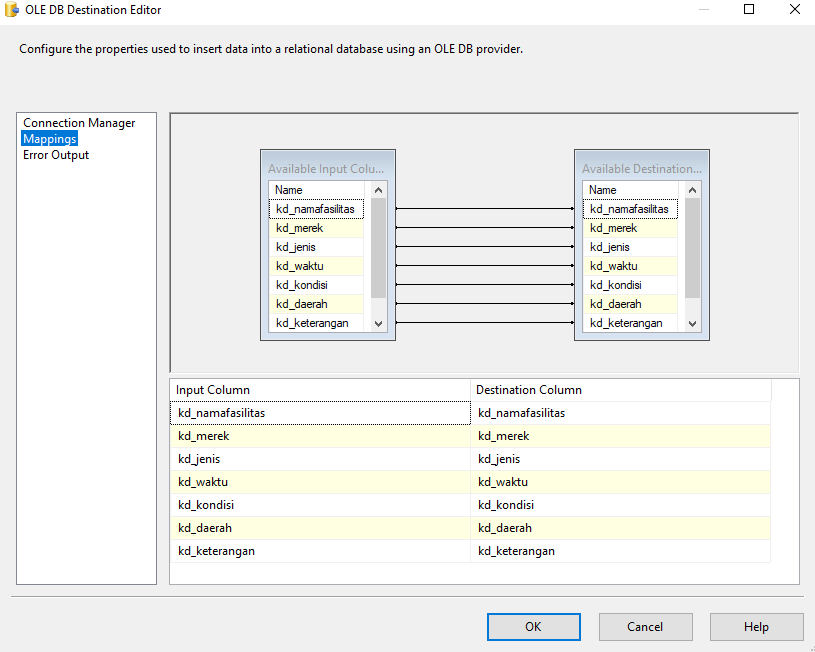
Kemudian klik double OLE DB *Destination*. Klik tanda panah pada OLE DB *connection manager*, lalu pilih *database* RGCPROSES. Kemudian pada *name of table or view* pilih [dbo].[fakta\_fasilitas] dan klik *preview*. Berikut ini dapat dilihat pada gambar 5.64 dibawah ini:

****

**Gambar 5.64 OLE DB Destination Fakta Fasilitas**

Pada gambar 5.64 merupakan SQL Command test pada tabel fakta fasilitas. Klik *preview* untuk mengetahui hasil dari SQL Command yang telah dengan benar.

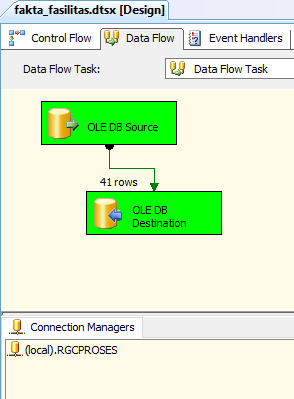
Kemudian klik *Mapping*. Untuk melihat hubungan data klik OK jika terhubung dan sesuai. Berikut ini dapat dilihat pada gambar 5.65 dibawah ini:



**Gambar 5.65 *Mapping* Fakta Fasilitas**

Pada gambar 5.65 merupakan tampilan *mapping* pada OLE DB *Destination*. Perhatikan tanda penghubung antara tabel dari input coloums dengan *destination coloums* terhubung dengan benar. Klik OK jika telah selesai.

Kemudian klik *start debugging* untuk memindahkan data ke dalam tabel fakta. Berikut ini dapat dilihat pada gambar 5.66 dibawah ini:



**Gambar 5.66 Berhasil Memindahkan Data ke Tabel Fakta Fasilitas**

Pada gambar 5.66 merupakan tampilan *debug* yang telah sukses dan data telah masuk kedalam database RGCPROSES.

Langkah selanjutnya adalah membuat fakta. Caranya adalah klik *new SSIS Package*, kemudian ganti namanya dengan fakta tenaga ahli. Kemudian klik *data flow*, pada *toolbox* tarik OLE DB *Source* dan OLE DB *Destination*, lalu tarik tanda panah dari OLE DB *Source* kearah OLE DB *Destination* untuk menghubungkannya. Klik *double* OLE DB *Source* lalu pada OLE DB *connection manager* pilih *new*, lalu pilih db\_proses kemudian klik OK. Pada *Data type* pilih *SQL* Command. Kemudian ketikan *query inner join* berikut pada kotak *query*:

***SELECT***

*dim\_tenagaahli.id\_tenagaahli,*

*dim\_daerah.kd\_daerah,*

*dim\_waktu.kd\_waktu,*

*dim\_tingkatpendidikan.kd\_tingkatpendidikan,*

*dim\_universitas.kd\_universitas,*

*dim\_jurusan.kd\_jurusan,*

*dim\_status.kd\_status,*

*AVG(proses\_tenagaahli.rata\_umur) AS Rata\_Umur*

***FROM proses\_tenagaahli***

*INNER JOIN dim\_tenagaahli*

*ON proses\_tenagaahli.TENAGA\_AHLI=dim\_tenagaahli.TENAGA\_AHLI*

*INNER JOIN dim\_daerah*

*ON proses\_tenagaahli.TEMPAT\_LAHIR=dim\_daerah.DAERAH*

*INNER JOIN dim\_waktu*

*ON proses\_tenagaahli.TANGGAL\_LAHIR=dim\_waktu.WAKTU*

*INNER JOIN dim\_tingkatpendidikan*

*ON proses\_tenagaahli.TINGKAT\_PENDIDIKAN=dim\_tingkatpendidikan.TINGKAT\_PENDIDIKAN*

*INNER JOIN dim\_universitas*

*ON proses\_tenagaahli.UNIVERSITAS=dim\_universitas.UNIVERSITAS*

*INNER JOIN dim\_jurusan*

*ON proses\_tenagaahli.JURUSAN=dim\_jurusan.JURUSAN*

*INNER JOIN dim\_status*

*ON proses\_tenagaahli.STATUS=dim\_status.STATUS*

***GROUP BY***

*dim\_tenagaahli.id\_tenagaahli,*

*dim\_daerah.kd\_daerah,*

*dim\_waktu.kd\_waktu,*

*dim\_tingkatpendidikan.kd\_tingkatpendidikan,*

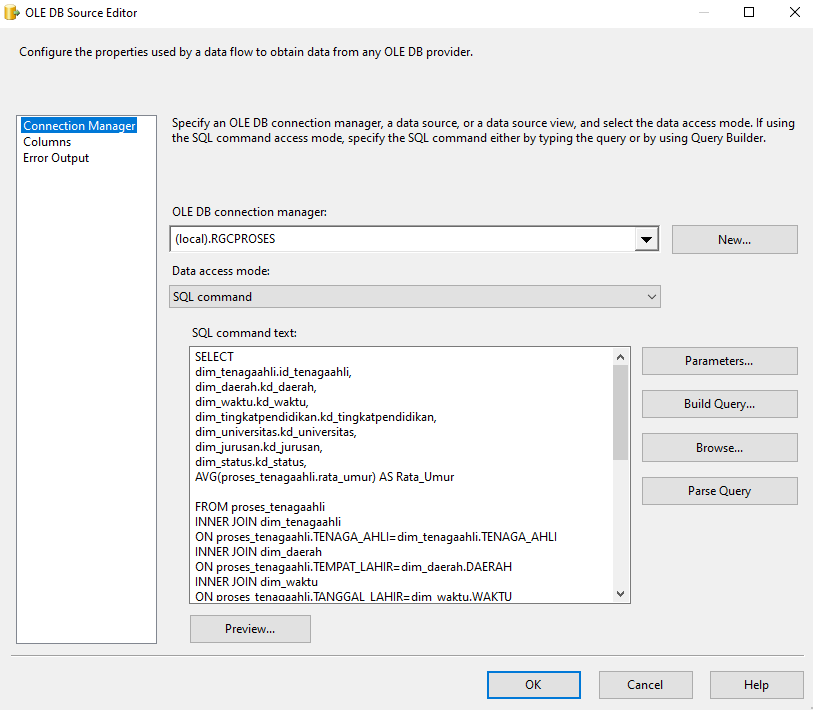
*dim\_universitas.kd\_universitas,*

*dim\_jurusan.kd\_jurusan,*

*dim\_status.kd\_status,*

*proses\_tenagaahli.rata\_umur*

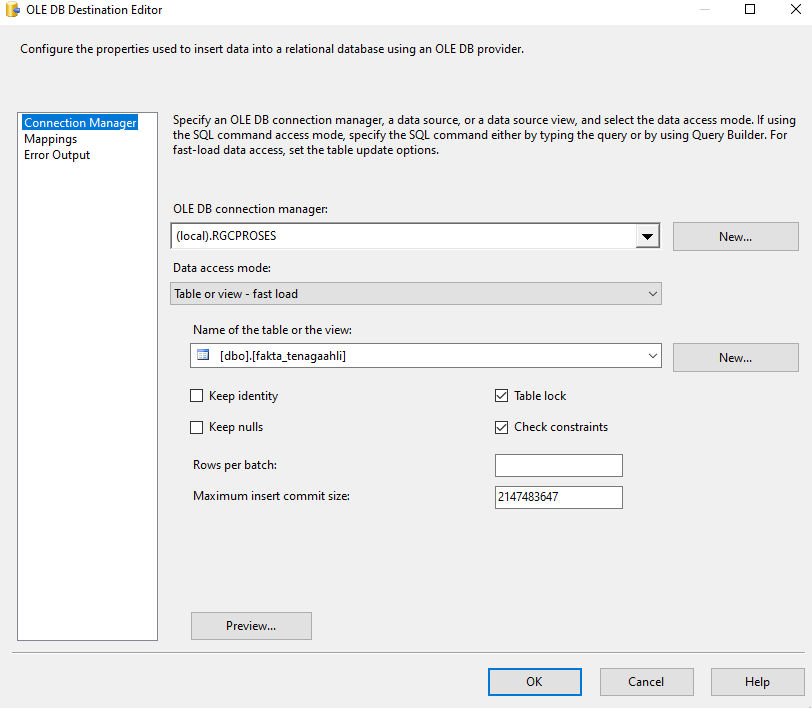
Kemudian klik *preview*, lalu OK. Berikut ini dapat dilihat pada gambar 5.67 dibawah ini:

****

**Gambar 5.67 OLE DB *Source* Fakta Tenaga Ahli**

Pada gambar 5.67 merupakan SQL Command test pada fakta tenaga ahli. Klik *preview* untuk mengetahui hasil dari SQL Command yang telah dengan benar.

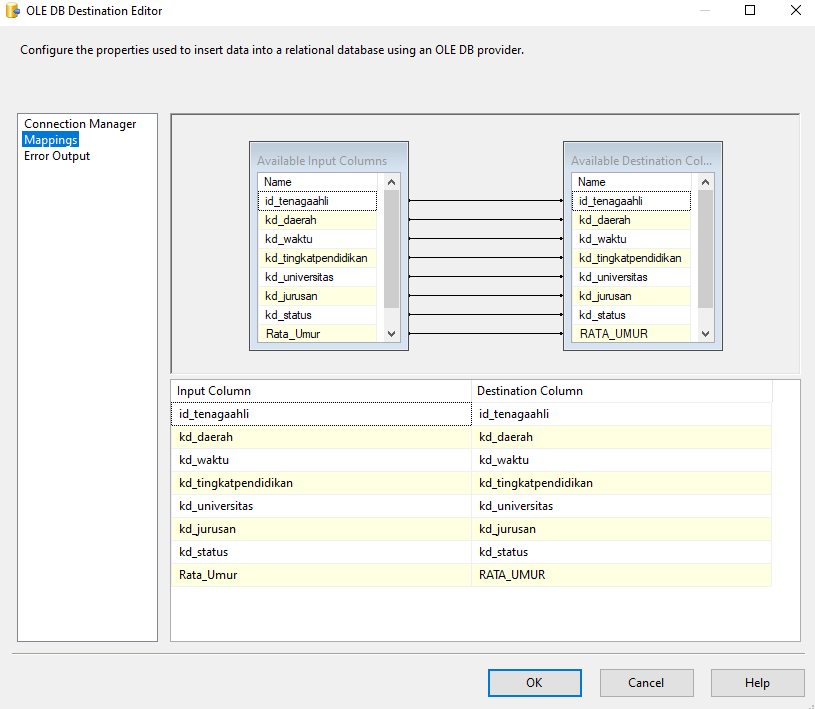
Kemudian klik double OLE DB *Destination*. Klik tanda panah pada OLE DB *connection manager*, lalu pilih *database* RGCPROSES. Kemudian pada *name of table or view* pilih [dbo].[fakta\_tenagaahli] dan klik *preview*. Berikut ini dapat dilihat pada gambar 5.68 dibawah ini:

****

**Gambar 5.68 OLE DB Destination Fakta Tenaga Ahli**

Pada gambar 5.68 merupakan tampilan utama setelah klik OLE DB *Destination*. OLE DB *Connection Manager* merupakan koneksi untuk memasukan data kedalam *database* sebagai media penyimpanan. Sesuaikan *name of the table view* dengan data yang telah diolah pada OLE DB *Source*.

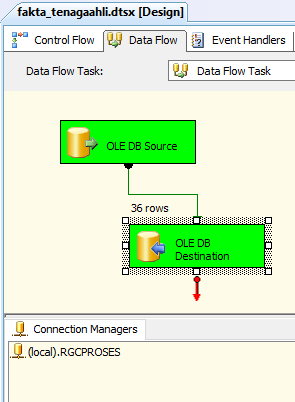
Kemudian klik *Mapping*. Untuk melihat hubungan data klik OK jika terhubung. Berikut ini dapat dilihat pada gambar 5.69 dibawah ini:



**Gambar 5.69 *Mapping* Fakta Tenaga Ahli**

Pada gambar 5.69 merupakan tampilan *mapping* pada OLE DB *Destination*. Perhatikan tanda penghubung antara tabel dari input coloums dengan *destination coloums* terhubung dengan benar. Klik OK jika telah selesai.

Kemudian klik start *debugging* untuk memindahkan data ke dalam tabel fakta. Berikut dapat dilihat pada gambar 5.70 dibawah ini:



**Gambar 5.70 Berhasil Memindahkan Data ke Tabel Fakta Tenaga Ahli**

Pada gambar 5.70 merupakan tampilan *debug* yang telah sukses dan data telah masuk kedalam database RGCPROSES.

Langkah selanjutnya adalah membuat fakta. Caranya adalah klik *new SSIS Package*, kemudian ganti namanya dengan fakta tenaga ahli. Kemudian klik *data flow*, pada *toolbox* tarik OLE DB *Source* dan OLE DB *Destination*, lalu tarik tanda panah dari OLE DB *Source* kearah OLE DB *Destination* untuk menghubungkannya. Klik *double* OLE DB *Source* lalu pada OLE DB *connection manager* pilih *new*, lalu pilih db\_proses kemudian klik OK. Pada *Data type* pilih *SQL* Command. Kemudian ketikan *query inner join* berikut pada kotak *query*:

***SELECT DISTINCT***

*dim\_instansi.kd\_instansi,*

*dim\_subbidang.kd\_subbidang,*

*dim\_tenagaahli.id\_tenagaahli,*

*dim\_daerah.kd\_daerah,*

*dim\_pemberijasa.id\_pemberijasa ,*

*dim\_waktu.kd\_waktu,*

*SUM (proses\_instansi.NILAI\_KONTRAK) AS NILAI\_KONTRAK*

***FROM proses\_instansi***

*INNER JOIN dim\_instansi*

*ON proses\_instansi.INSTANSI=dim\_instansi.INSTANSI*

*INNER JOIN dim\_subbidang*

*ON proses\_instansi.SUB\_BIDANG=dim\_subbidang.SUB\_BIDANG*

*INNER JOIN dim\_tenagaahli*

*ON proses\_instansi.TENAGA\_AHLI=dim\_tenagaahli.TENAGA\_AHLI*

*INNER JOIN dim\_daerah*

*ON proses\_instansi.LOKASI=dim\_daerah.DAERAH*

*INNER JOIN dim\_pemberijasa*

*ON proses\_instansi.PEMBERI\_JASA=dim\_pemberijasa.PEMBERI\_JASA*

*INNER JOIN dim\_waktu*

*ON proses\_instansi.KONTRAK\_INSTANSI=dim\_waktu.WAKTU*

***GROUP BY***

*dim\_instansi.kd\_instansi,*

*dim\_subbidang.kd\_subbidang,*

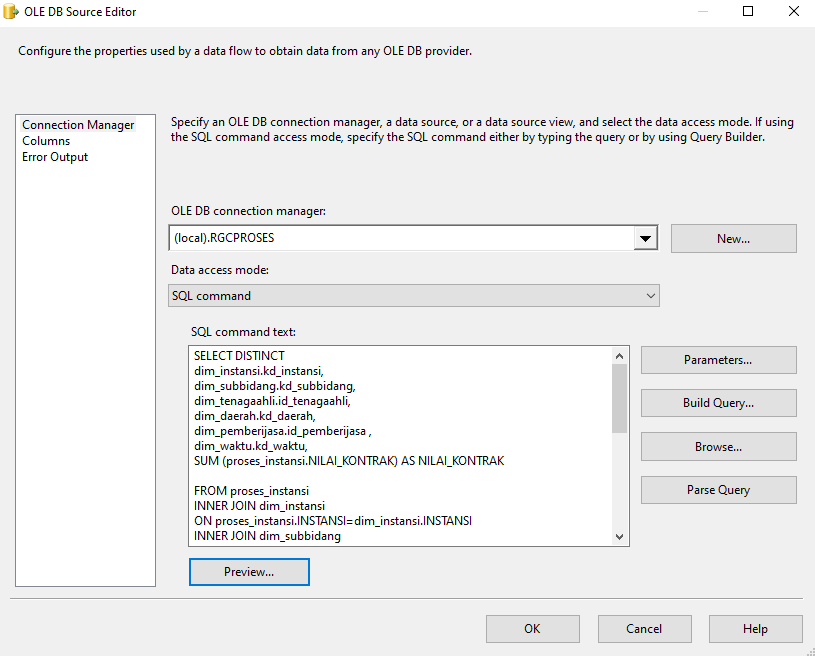
*dim\_tenagaahli.id\_tenagaahli,*

*dim\_daerah.kd\_daerah,*

*dim\_pemberijasa.id\_pemberijasa,*

*dim\_waktu*

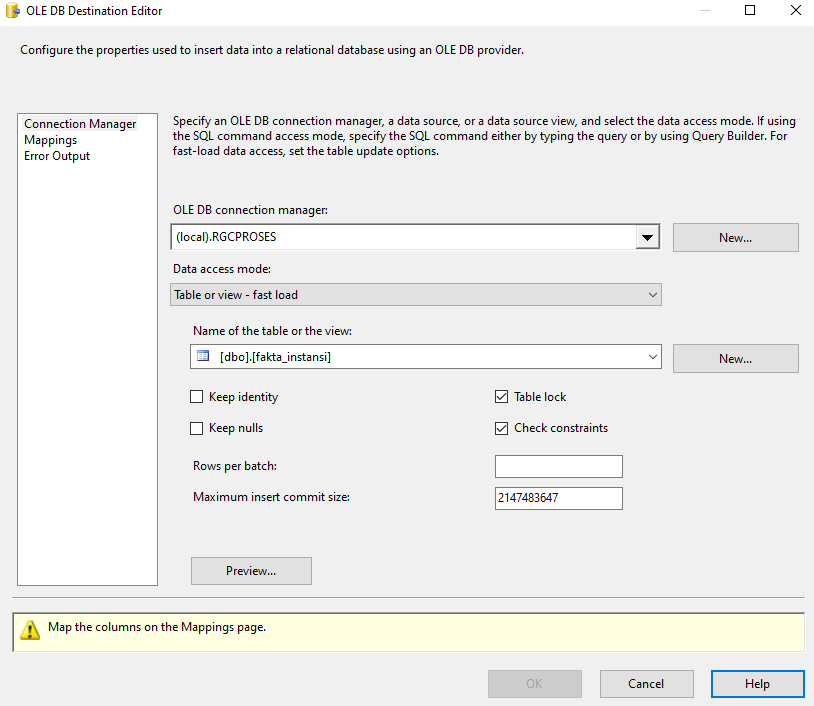
Kemudian klik *preview*, lalu OK. Berikut dapat dilihat pada gambar 5.71 dibawah ini:

****

**Gambar 5.71 OLE DB *Source* Fakta Instansi**

Pada gambar 5.71 merupakan SQL Command test pada fakta instansi. Klik *preview* untuk mengetahui hasil dari SQL Command yang telah dengan benar.

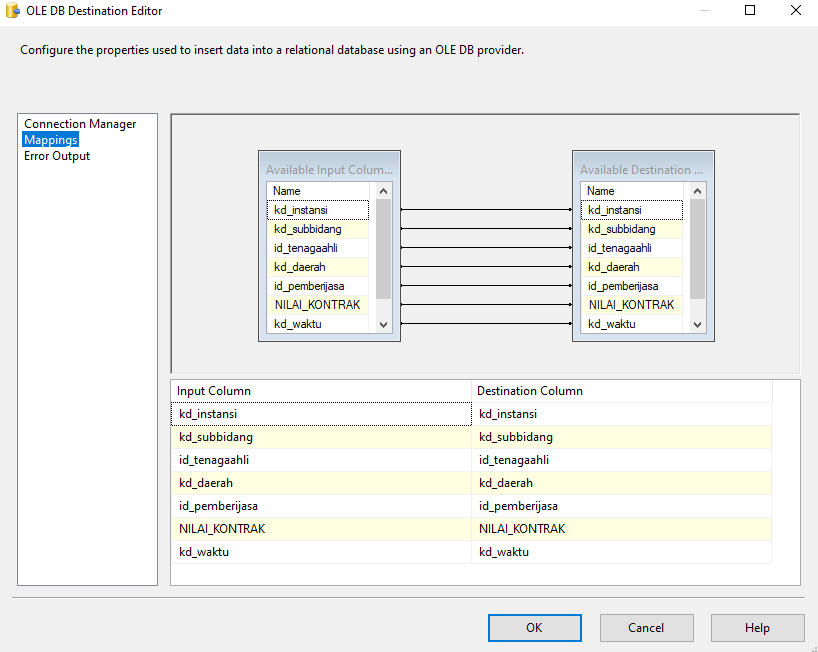
Kemudian klik double OLE DB *Destination*. Klik tanda panah pada OLE DB *connection manager*, lalu pilih *database* RGCPROSES. Kemudian pada *name of table or view* pilih [dbo].[fakta\_instansi] dan klik *preview*. Berikut dapat dilihat pada gambar 5.70 dibawah ini:

****

**Gambar 5.72 OLE DB Destination Fakta Instansi**

Pada gambar 5.72 merupakan tampilan utama setelah klik OLE DB *Destination*. OLE DB *Connection Manager* merupakan koneksi untuk memasukan data kedalam *database* sebagai media penyimpanan. Sesuaikan *name of the table view* dengan data yang telah diolah pada OLE DB *Source*.

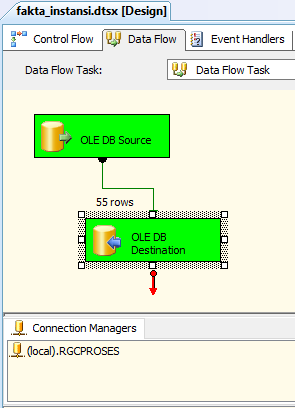
Kemudian klik *Mapping*. Untuk melihat hubungan data klik OK jika terhubung. Berikut dapat dilihat pada gambar 5.73 dibawah ini:



**Gambar 5.73 *Mapping* Fakta Instansi**

Pada gambar 5.73 merupakan tampilan *mapping* pada OLE DB *Destination*. Perhatikan tanda penghubung antara tabel dari input coloums dengan *destination coloums* terhubung dengan benar. Klik OK jika telah selesai.

Kemudian klik *start debugging* untuk memindahkan data ke dalam tabel fakta. Berikut dapat dilihat pada gambar 5.74 dibawah ini:



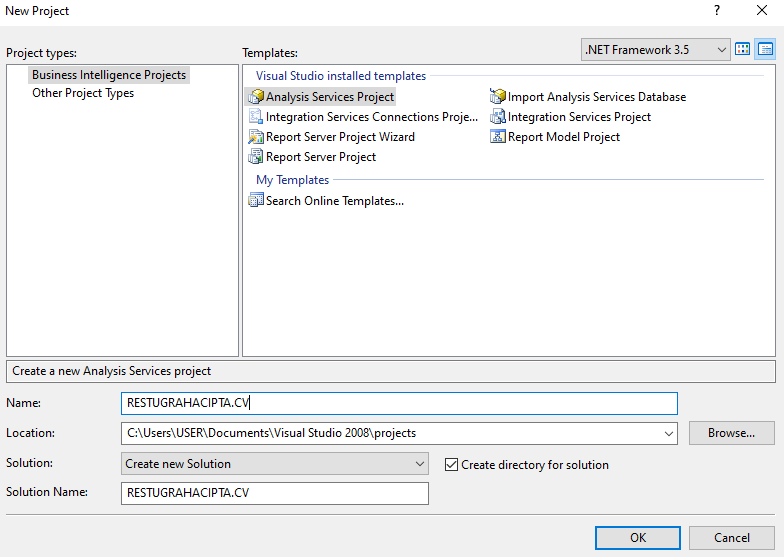
**Gambar 5.74 Berhasil Memindahkan Data ke Tabel Fakta Instansi**

Pada gambar 5.74 merupakan tampilan *debug* yang telah sukses dan data telah masuk kedalam database RGCPROSES.

# Implementasi *Analysis Services*

*SQL Server Analysis services* (SSAS) merupakan proses penyajian data dan analisis data.

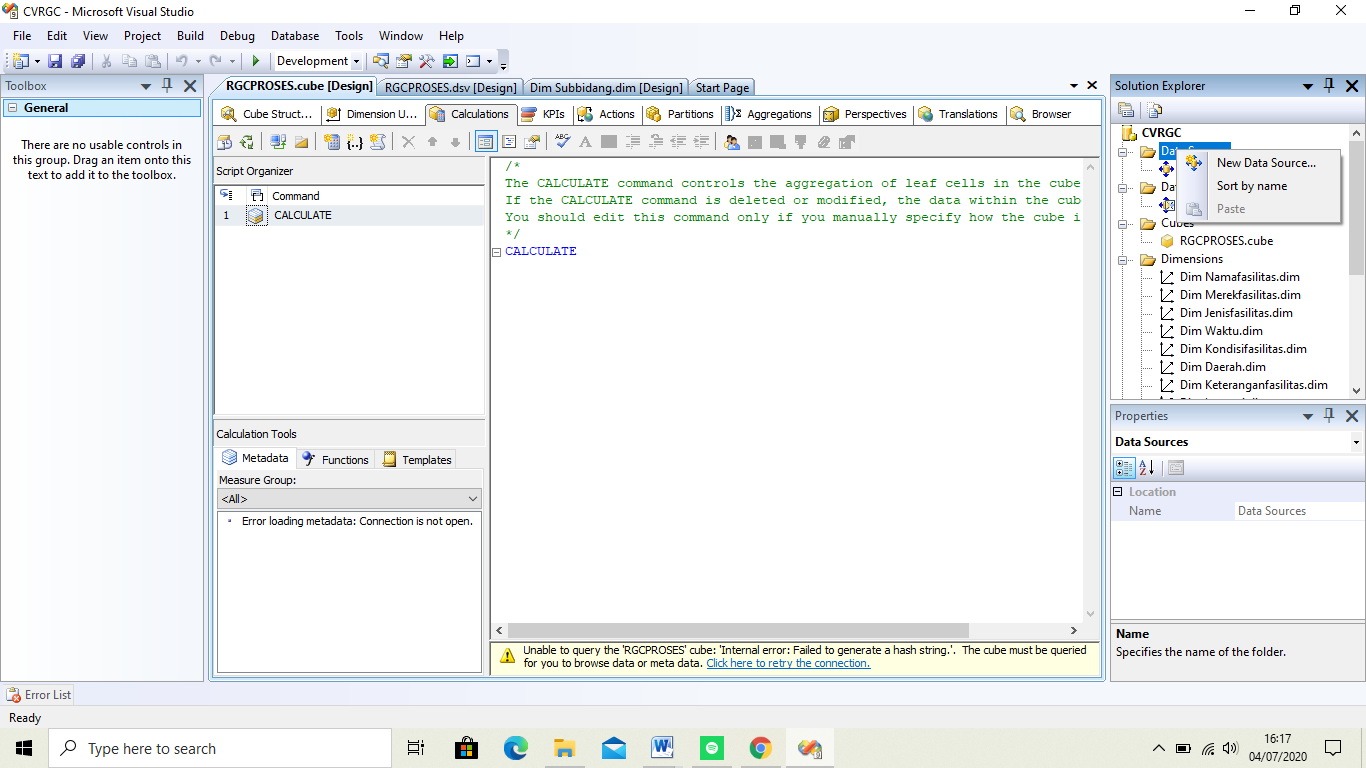
Langkah awal membuat *analysis service* untuk *data warehouse* ini. Buka program *SQL Server Business Interlligence Development Studio*. Klik file, pilih new, lalu project. Kemudian pilih *analysis services project*, lalu ubah namA dengan analysis RESTUGRAHACIPTA.CV. Pilih *location* untuk lokasi penyimpanan klik OK. Berikut dapat dilihat pada gambar 5.75 dibawah ini:



**Gambar 5.75 *Analysis Services Project***

Pada gambar 5.75 merupakan tampilan menu pilihan SQL *Server Business Intelligence*. Pilih *Analysis Services Project* untuk melakukan analisis terhadap data.

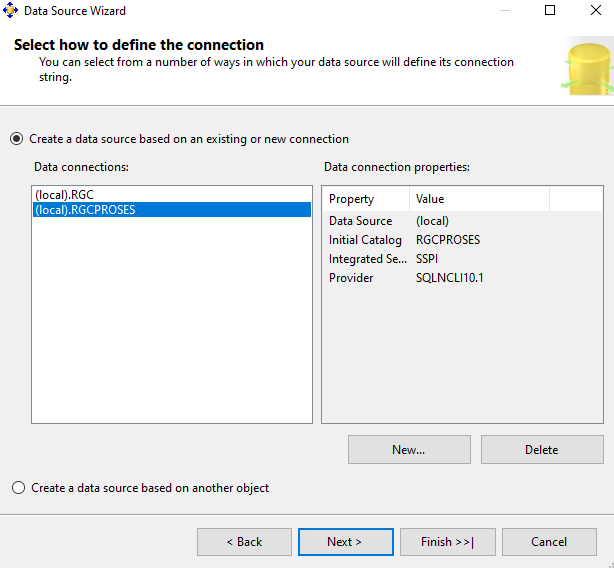
Selanjutnya klik kanan pada *Data Source*, pilih *new Data Source*. Berikut dapat dilihat pada gambar 5.76 dibawah ini:



**Gambar 5.76 *New Data Source Analysis***

Pada gambar 5.76 setelah klik OK pada tahap selanjutnya, lakukan pembuatan *data source*.

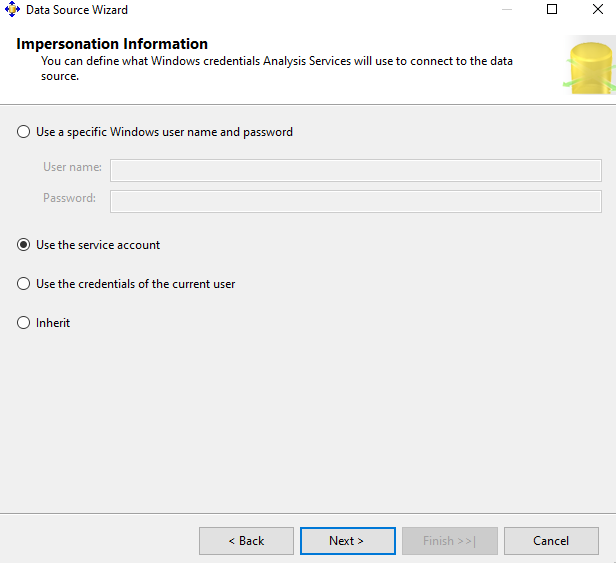
Kemudian akan muncul *Data Source Wizard* klik *next*. Setelah itu pada *Select how to define the connection*, pilih *database* RGCPROSES, lalu klik *next*. Berikut dapat dilihat pada gambar 5.77 dibawah ini:



**Gambar 5.77 *Select how to define the connection***

Pada gambar 5.77 proses dalam melakukan koneksi, *select Create a data source based on an exixting or new connection* lalu pilih new untuk menambahkan koneksi pada *data source*.

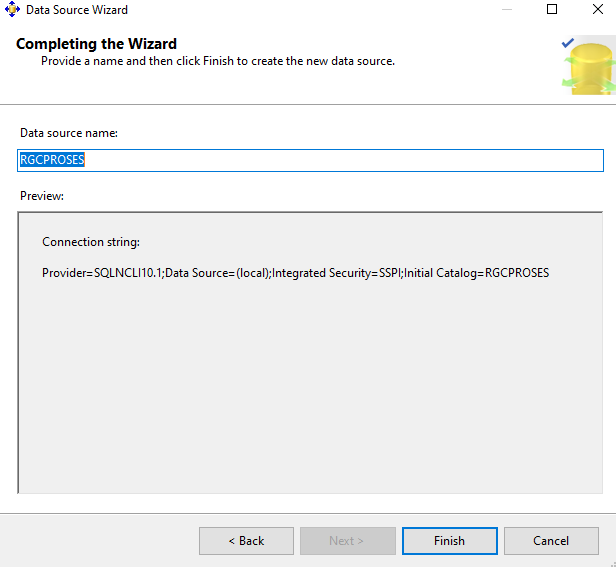
Selanjutnya tahap *impersonation information*. Tahap ini bertujuan untuk memberikan *password* terhadap *analysis service*. Pilih *use the service account*, maksudnya adalah menggunakan *password* dari *widows server* dan klik *next*. Berikut dapat dilihat pada gambar 5.78 dibawah ini:



**Gambar 5.78 *Impersonation Information***

Pada gambar 5.78 proses dalam pemilihan kategori *impersonation information*. Klik *use the service account* dan klik *next*.

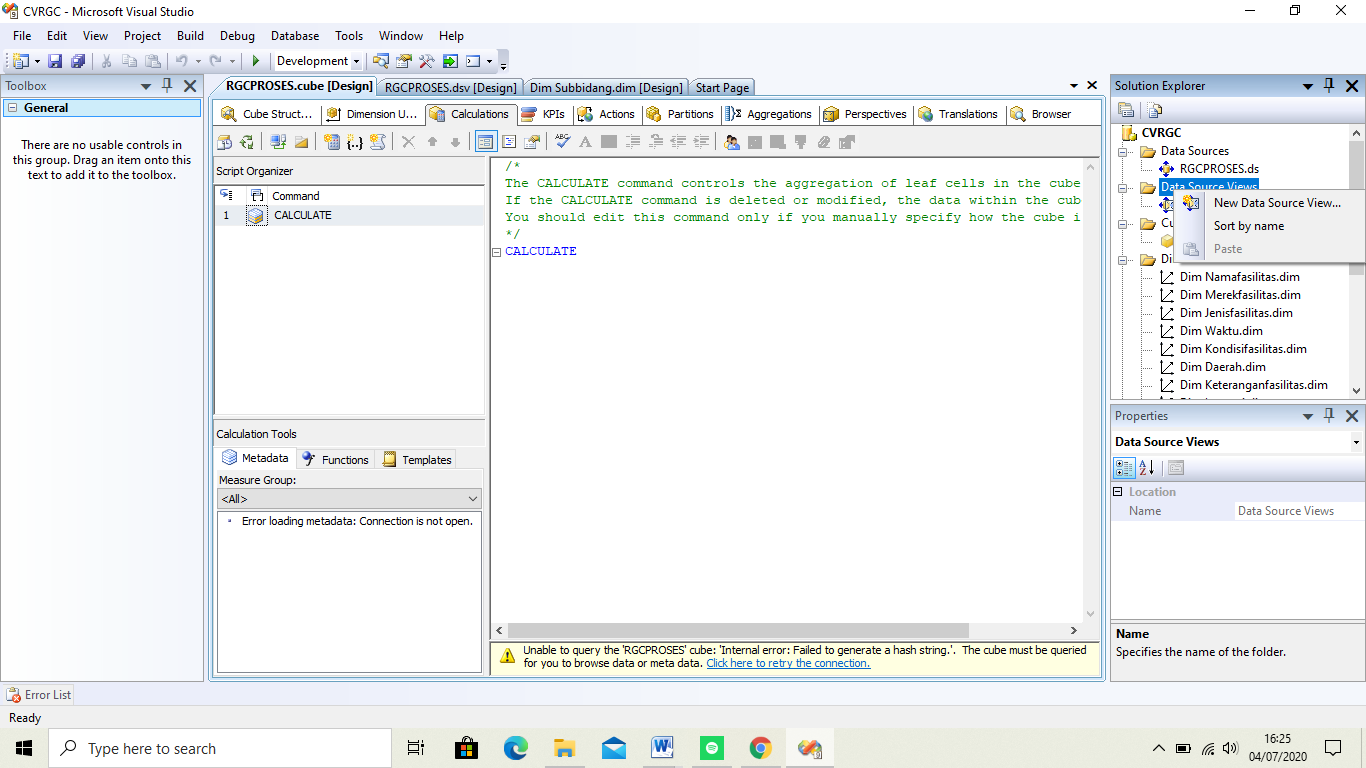
Kemudian akan muncul *complete wizard* dan klik *Finish*. Berikut dapat dilihat pada gambar 5.79 dibawah ini:



**Gambar 5.79 *Complete wizard Analysis Service***

Pada gambar 5.79 merupakan tampilan dari *completing the wizard* untuk memastikan *data source* yang dipilih telah tepat.

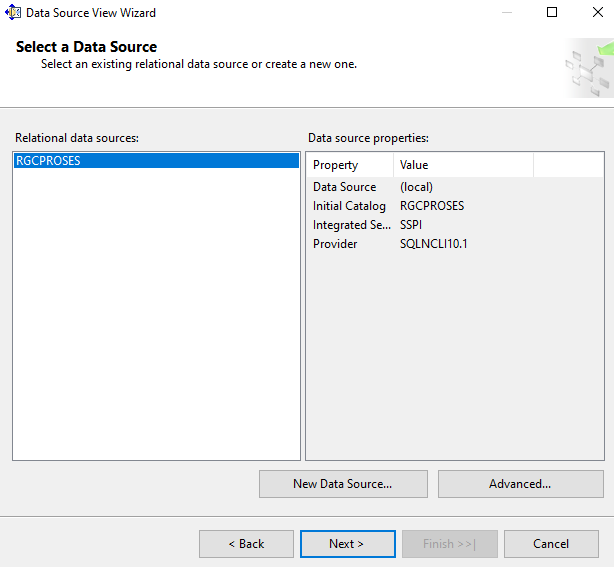
Selanjutnya klik kanan pada *data source view*, klik *new data source views.* Berikut dapat dilihat pada gambar 5.80 dibawah ini:



**Gambar 5.80 *New Data Source Views***

Pada gambar 5.80 proses dalam membuat data source views . klik kanan untuk membuat data source view.

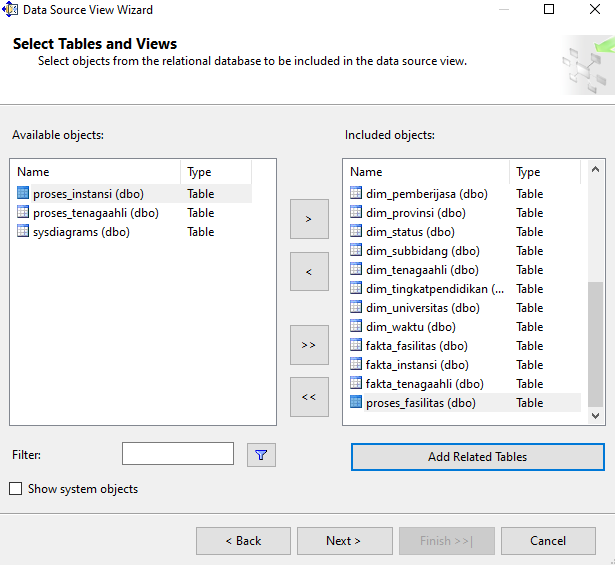
Kemudian muncul *data source view wizard* klik *next*. Pada *select a data source*, pilih Db Proses klik *next*. Berikut dapat dilihat pada gambar 5.81 dibawah ini:



**Gambar 5.81 *Select a Data Source***

Pada gambar 5.81 merupakan koneksi database dari data source views. Sesuaikan koneksi database data source dengan data source views

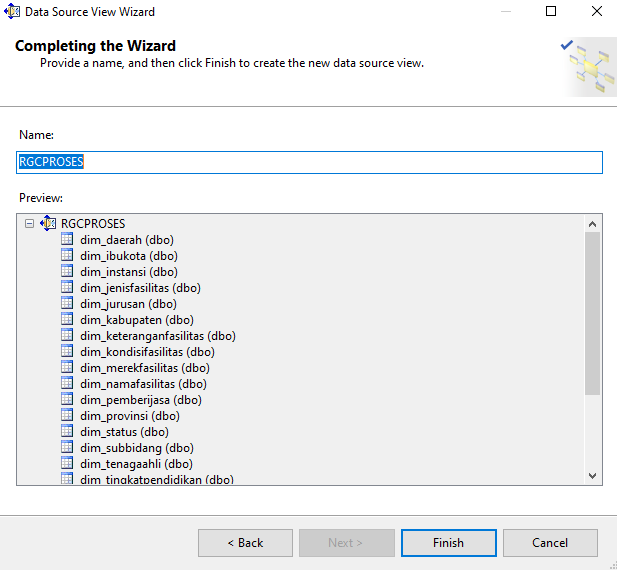
Kemudian akan muncul *select table and views*, pilih dbo.fakta\_penjualan lalu klik tombol >, lalu klik *add related table*, maka tabel-tabel yang berelasi dengan tabel fakta\_penjualan akan pindah ke kanan dan klik *next*. Berikut dapat dilihat pada gambar 5.82 dibawah ini:



**Gambar 5.82 *Select Tables and Views***

Pada gambar 5.82 proses dalam pemilihan tabel yang akan dianalisa kedalam analisis service project. Setelah memilih tabel berdasarkan kebutuhan klik next untuk ke tahap selanjutnya.

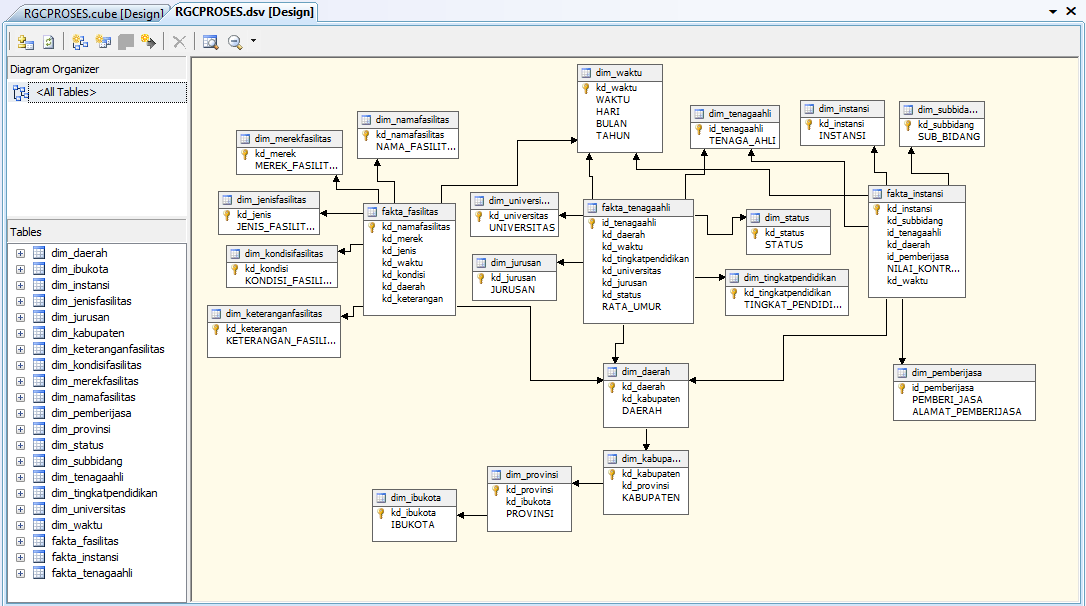
Selanjutnya akan muncul *complete wizard* dan klik *finish*. Berikut dapat dilihat pada gambar 5.83 dibawah ini:

****

**Gambar 5.83 *Complete Wizard Data Source View***

Pada gambar 5.83 proses completing the wizard untuk memeriksa kembali tabel yang dipilih berdasarkan kebutuhan dan klik finish jika tabel yang dipilih telah sesuai.

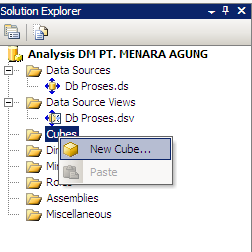
Setelah itu akan muncul tampilan relasi antar tabel seperti gambar 5.84 berikut:



**Gambar 5.84 Tampilan Relasi antar Tabel**

Pada gambar 5.84 merupakan struktur relasi antar tabel yang telah dipilih, tabel tersebut terdiri dari dimensi waktu, tenaga ahli, tingkat pendidikan, jurusan, universitas, instansi, sub bidang, pemberi jasa, nama fasilitas, jenis fasilitas, kondisi fasilitas, keterangan fasilitas, status, pada dimensi daerah, kabupaten, provinsi, ibukota memiliki struktur skema snowflake yang mana antar dimensi saling berelasi satu sama lain dan terdapat fakta instansi, tenaga ahli dan fasilitas.

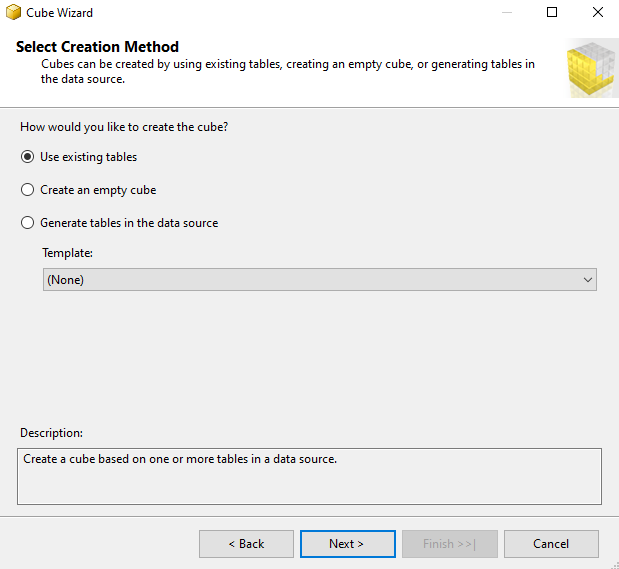
Langkah selanjutnya klik kanan pada *cube* dan klik *new cube*. Seperti gambar 5.85 berikut:

****

**Gambar 5.85 *New Cube***

Pada gambar 5.85 proses pemilihan cube merupakan proses data yang telah digabungkan untuk memudahkan *query* dalam mengembalikan data secara cepat.

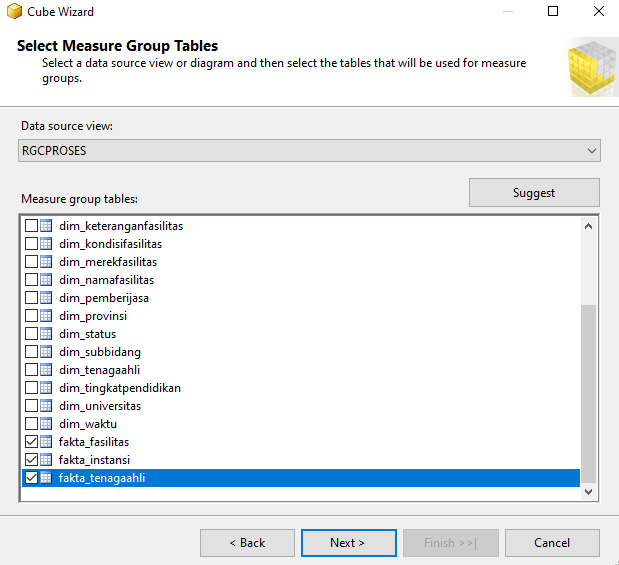
Kemudian akan muncul *cube wizard* maka klik *next*. Lalu pada *select creation method* pilih *use existing tables* dan klik *next*. Berikut dapat dilihat pada gambar 5.86 dibawah ini:



**Gambar 5.86 *Select Build Method Cube***

Pada gambar 5.86 merupakan proses untuk memilih *creation method*. Klik next setelah memilih *creation method* pada cube.

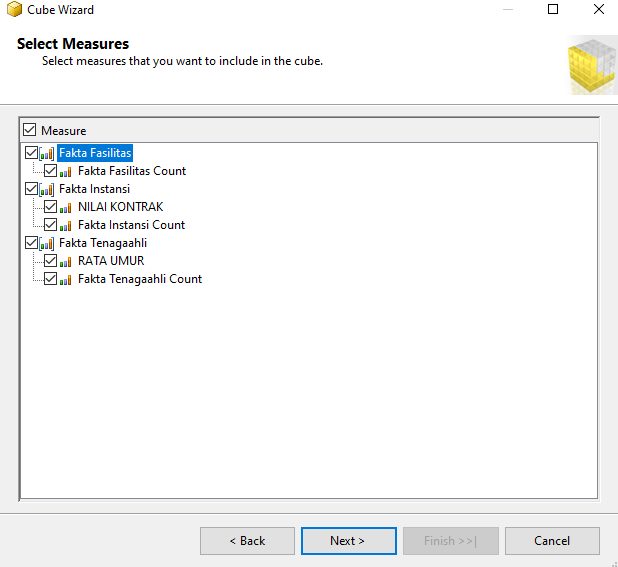
Lalu akan muncul *select measure group tables* dan pilih tabel fakta yang akan ditampilkan. Berikut dapat dilihat pada gambar 5.87 dibawah ini:



**Gambar 5.87 *Select Data Source View Cube***

Pada gambar 5.87 merupakan proses dalam pemilihan *measure* untuk melakukan perhitungan, select table fakta fasilitas, fakta instansi, dan fakta tenaga ahli.

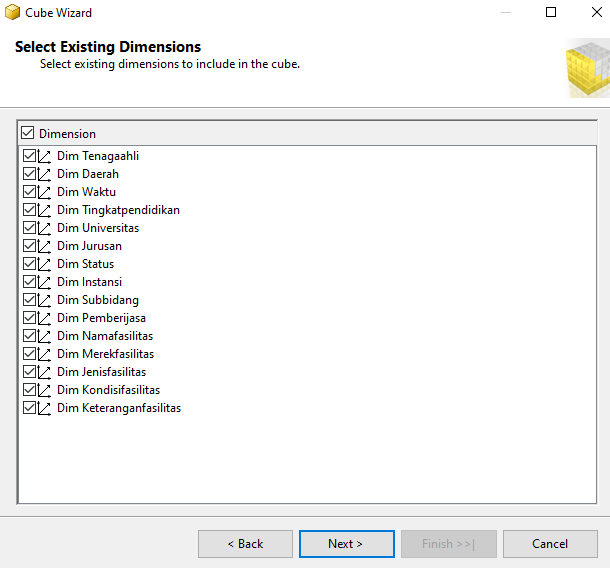
Kemudian akan melakukan *fact tabel* secara otomatis seperti gambar 5.88 berikut dan klik *next*.



**Gambar 5.88 *Detecting Fact Tables***

Pada gambar 5.88 merupakan tampilan dari *measure* yang telah dipilih pada tahap sebelumnya.

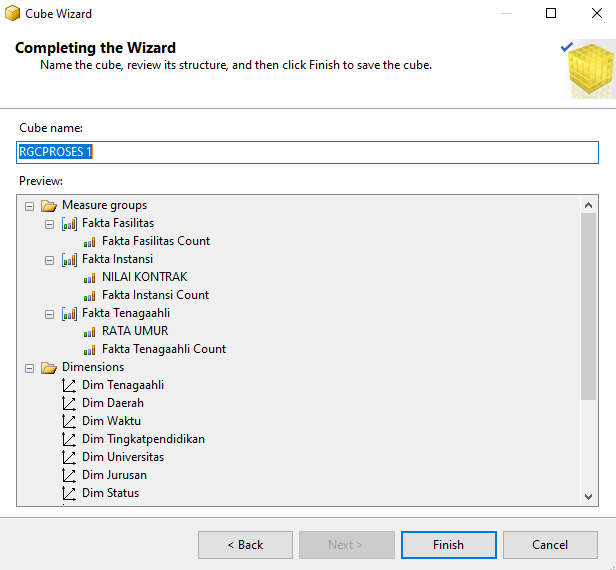
Lalu akan muncul *select existing dimension*. Lakukan deteksi dimensi tabel secara otomatis lalu klik next. Berikut dapat dilihat pada gambar 5.89 dibawah ini:



**Gambar 5.89 *Detecting Dimension Tables***

Pada gambar 5.89 merupakan tampilan dari seluruh dimensi yang telah dipilih pada tahap sebelumya.

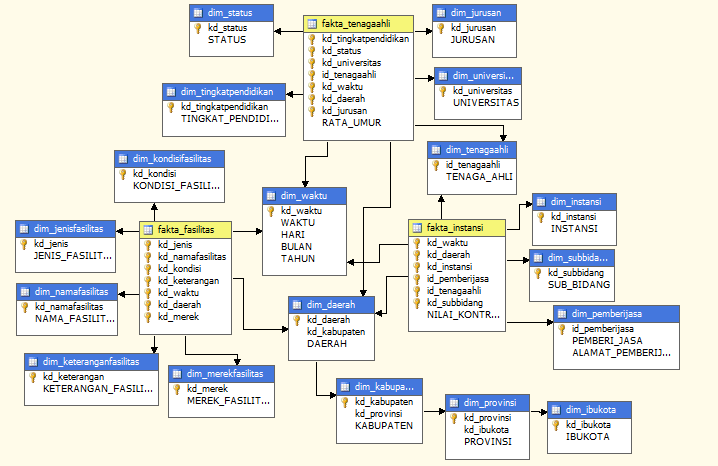
Setelah itu akan masuk ke *completing the wizard* untuk memastikan *fact and dimension table* sesuai dengan pilihan dan klik *finish* seperti gambar 5.90 berikut:



**Gambar 5.90 *Identify Fact and Dimension Table***

Pada gambar 5.90 merupakan tampilan *completing the wizard*. Untuk melihat kembali dimensi serta *measure* yang telah dipilih benar.

Setelah menekan tombol *finish* maka akan muncul struktur *cube* yang telah dipilih seperti gambar 5.91 dibawah ini :



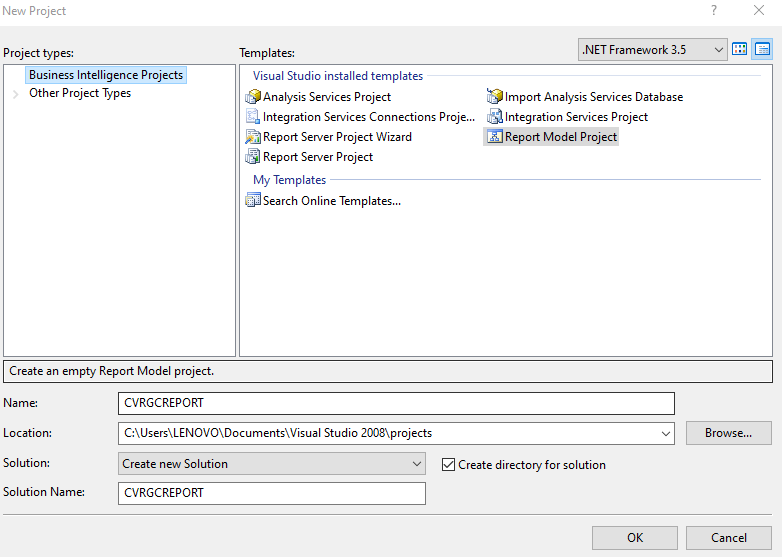
**Gambar 5.91 *Structur Cube***

Pada gambar 5.91 merupakan struktur data pada *cube*. Tabel tersebut terdiri dari dimensi waktu, tenaga ahli, tingkat pendidikan, jurusan, universitas, instansi, sub bidang, pemberi jasa, nama fasilitas, jenis fasilitas, kondisi fasilitas, keterangan fasilitas, status, pada dimensi daerah, kabupaten, provinsi, ibukota memiliki struktur skema snowflake yang mana antar dimensi saling berelasi satu sama lain dan terdapat fakta instansi, tenaga ahli dan fasilitas berisi *key-key* dari dimensi yang saling terhubung ke tabel fakta.

# Implementasi Reporting Services

*SQL Server Reporting Services* (SSRS) merupakan bagian *platform* laporan yang berbasis server dapat digunakan untuk menyediakan fungsionalitas dalam pembuatan laporan dari sumber data. SSRS memudahkan dalam pembuatan laporan dan dapat mengintegrasikan laporan dengan berbagai pengaturan dan juga dapat digunakan untuk menampilkan data dari database.

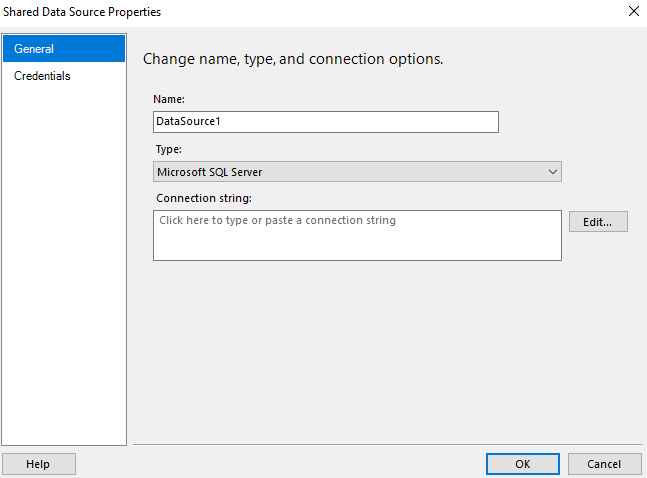
Pada SQL *Server Business Intelligence Development* *Studio* membuat *reporting server projec*t. Klik *new project* pilih *reporting server project*, ganti nama dengan RGCREPORT dan pilih tempat penyimpanan untuk menyimpan project klik OK. Berikut dapat dilihat pada gambar 5.92 dibawah ini:



**Gambar 5.92 Tampilan *New Project***

Pada gambar 5.92 tampilan dari SQL *Server Business Intelligence Project* untuk membuat *platform* laporan pada *report model project*. Klik OK setelah membuat nama *project*, memilih lokasi penyimpanan *project*.

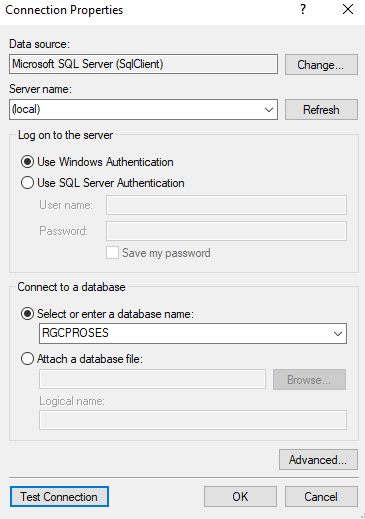
Setelah masuk kedalam *project*, klik kanan pada *Shared Data Source* dan klik *Add New Data Source*. Berikut dapat dilihat pada gambar 5.93 dibawah ini:



**Gambar 5.93 Tampilan *Shared Data Source***

Pada gambar 5.93 merupakan tampilan untuk membuat data source pada report.

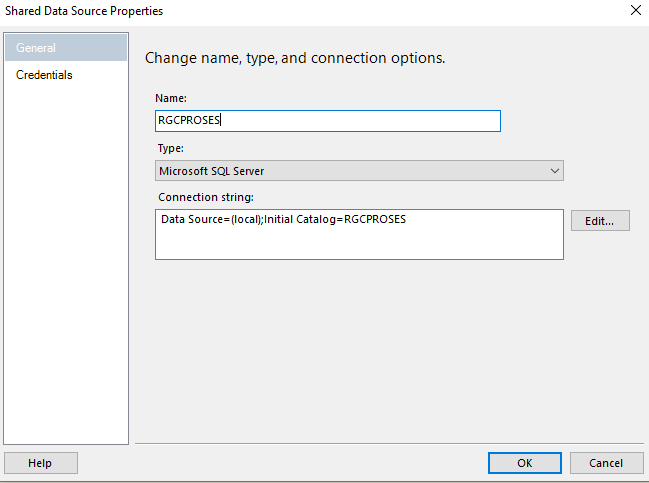
Kemudian klik *edit* pilih *connection properties* yang akan dihubungkan pada *project reporting* lalu klik *test connection* untuk memastikan koneksi dapat terhubung dan klik OK. Berikut dapat dilihat pada gambar 5.94 dibawah ini:



**Gambar 5.94 Tampilan *Connection Properties***

Pada gambar 5.94 merupakan tampilan dari *connection properties*, ubah *server name* sesuai dengan *server name* yang digunakan pada SQL *Server Management*, pada koneksi database *select* RGCPROSES yang merupakan penyimpanan data untuk dibuat laporan.

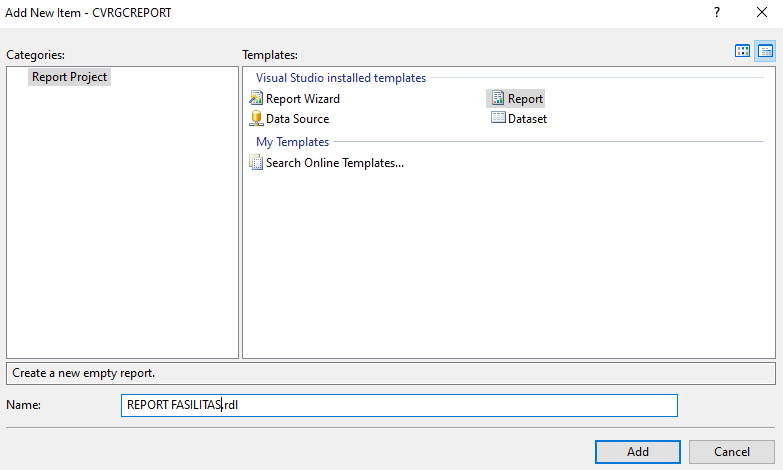
Langkah selanjutnya ganti nama pada gambar dibawah dengan *data source* yang telah dipilih sebelumnya agar dapat terkoneksi dan terbaca kemudian klik OK. Berikut dapat dilihat pada gambar 5.95 dibawah ini:



**Gambar 5.95 Tampilan *Shared Data Source Properties***

Pada gambar 5.95 merupakan tampilan properties dari *shared* *data source*.

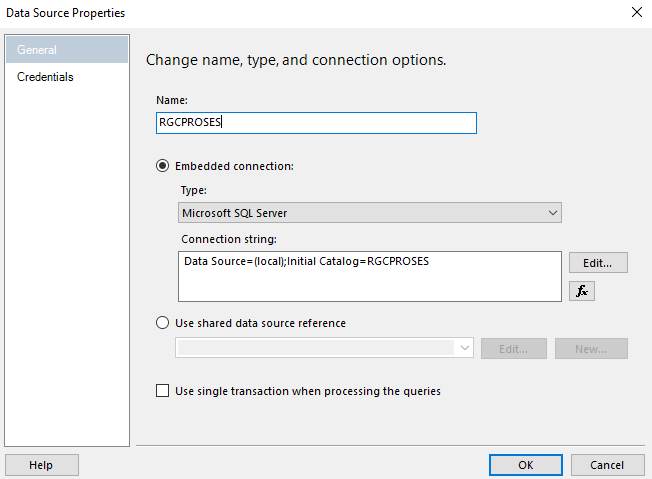
Kemudian klik kanan pada *Shared Dataset* klik *add* dan pilih *report*. Ganti nama untuk REPORT FASILITAS yang akan di tampilkan. Klik *add* seperti gambar 5.96 dibawah ini:



**Gambar 5.96 Tampilan *Report Project* Fasilitas**

Pada gambar 5.96 merupakan tampilan dari *properties report*. Buat nama berdasarkan laporan yang akan dibuat dan klik *Add*.

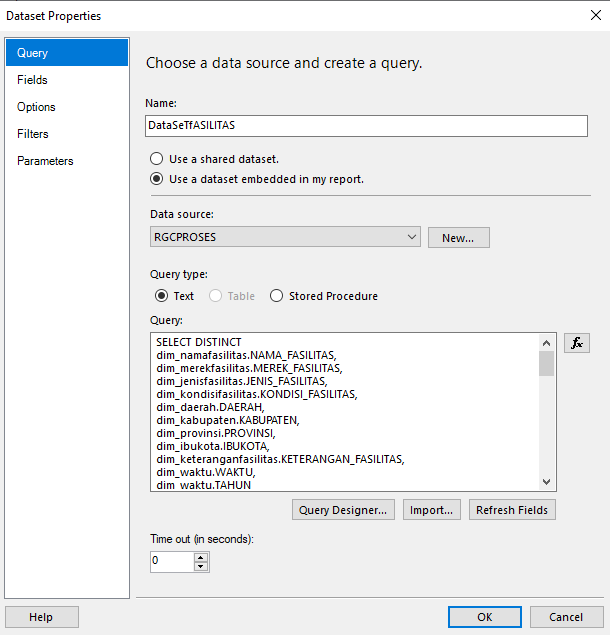
Setelah klik *add* akan muncul tampilan *report* data pada *project* yang akan dibuat. Klik *data source* sesuaikan data *source project* dengan *data source* yang ada pada *reporting project* dan klik OK. Berikut dapat dilihat pada gambar 5.97 dibawah ini:



**Gambar 5.97 Tampilan *Data Source Properties***

Pada gambar 5.97 proses membuat data source, pada nama ubah dengan database RGCPROSES dengan data source (*local*) dan klik OK.

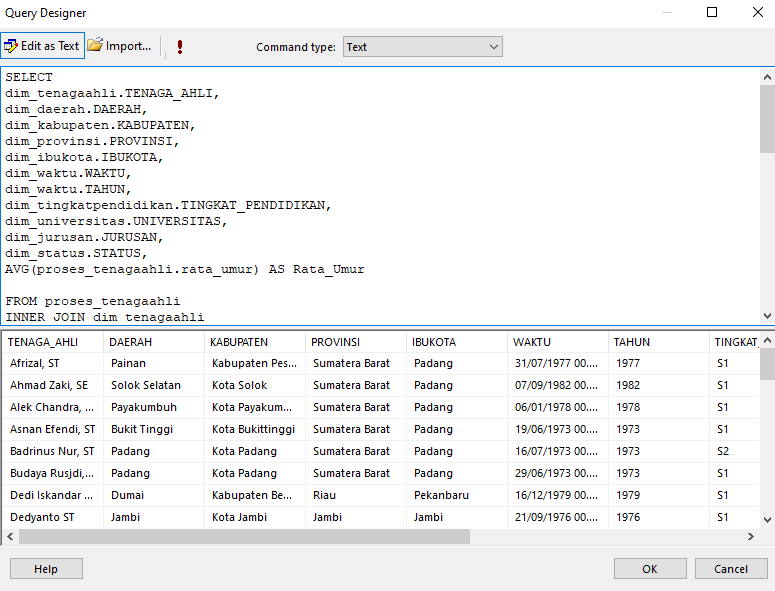
Kemudian klik kanan *dataset* dan pilih add *dataset*, ganti nama data set dengan nama report, pilih *use a data set embedded in my report*. Pilih *data source* sebelumnya dengan *query* type text dan klik *query* designer. Berikut dapat dilihat pada gambar 5.98 dibawah ini:



**Gambar 5.98 Tampilan *Data Set Properties* Fasilitas**

Pada gambar 5.98 tampilan *query* dari *dataset proeprties*. Ubah nama *dataset* menjadi datasetFASILITAS, pilih data *source* RGCPROSES dimana data yang dibutuhkan tersimpan.

Setelah itu klik *query designer* buat *query* yang akan menampilkan tampilan report. Klik run (!) untuk menjalankan *query* yang telah dibuat dan tidak ada error klik OK. Berikut dapat dilihat pada gambar 5.99 dibawah ini:



**Gambar 5.99 Tampilan *Query Designer* Fasilitas**

Pada gambar 5.99 proses untuk memasukan query pada laporan fakta fasilitas.

*Query* fasilitas :

***SELECT DISTINCT***

*dim\_namafasilitas.NAMA\_FASILITAS,*

*dim\_merekfasilitas.MEREK\_FASILITAS,*

*dim\_jenisfasilitas.JENIS\_FASILITAS,*

*dim\_kondisifasilitas.KONDISI\_FASILITAS,*

*dim\_daerah.DAERAH,*

*dim\_kabupaten.KABUPATEN,*

*dim\_provinsi.PROVINSI,*

*dim\_ibukota.IBUKOTA,*

*dim\_keteranganfasilitas.KETERANGAN\_FASILITAS,*

*dim\_waktu.WAKTU,*

*dim\_waktu.TAHUN*

***FROM proses\_fasilitas***

*INNER JOIN dim\_namafasilitas*

*ON proses\_fasilitas.NAMA\_FASILITAS=dim\_namafasilitas.NAMA\_FASILITAS*

*INNER JOIN dim\_merekfasilitas*

*ON proses\_fasilitas.MEREK\_FASILITAS=dim\_merekfasilitas.MEREK\_FASILITAS*

*INNER JOIN dim\_jenisfasilitas*

*ON proses\_fasilitas.JENIS\_FASILITAS=dim\_jenisfasilitas.JENIS\_FASILITAS*

*INNER JOIN dim\_kondisifasilitas*

*ON proses\_fasilitas.KONDISI\_FASILITAS=dim\_kondisifasilitas.KONDISI\_FASILITAS*

*INNER JOIN dim\_waktu*

*ON proses\_fasilitas.WAKTU\_PEMBELIAN=dim\_waktu.WAKTU*

*INNER JOIN dim\_keteranganfasilitas*

*ON proses\_fasilitas.KETERANGAN\_FASILITAS=dim\_keteranganfasilitas.KETERANGAN\_FASILITAS*

*INNER JOIN dim\_daerah*

*ON proses\_fasilitas.LOKASI\_FASILITAS=dim\_daerah.DAERAH*

*INNER JOIN dim\_kabupaten*

*ON dim\_daerah.kd\_kabupaten=dim\_kabupaten.kd\_kabupaten*

*INNER JOIN dim\_provinsi*

*ON dim\_kabupaten.kd\_provinsi=dim\_provinsi.kd\_provinsi*

*INNER JOIN dim\_ibukota*

*ON dim\_provinsi.kd\_ibukota=dim\_ibukota.kd\_ibukota*

***GROUP BY***

*dim\_namafasilitas.NAMA\_FASILITAS,*

*dim\_merekfasilitas.MEREK\_FASILITAS,*

*dim\_jenisfasilitas.JENIS\_FASILITAS,*

*dim\_kondisifasilitas.KONDISI\_FASILITAS,*

*dim\_daerah.DAERAH,*

*dim\_kabupaten.KABUPATEN,*

*dim\_provinsi.PROVINSI,*

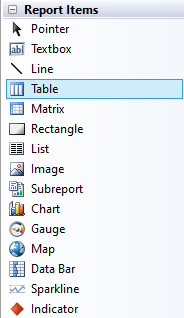
*dim\_ibukota.IBUKOTA,*

*dim\_keteranganfasilitas.KETERANGAN\_FASILITAS,*

*dim\_waktu.WAKTU,*

*dim\_waktu.TAHUN*

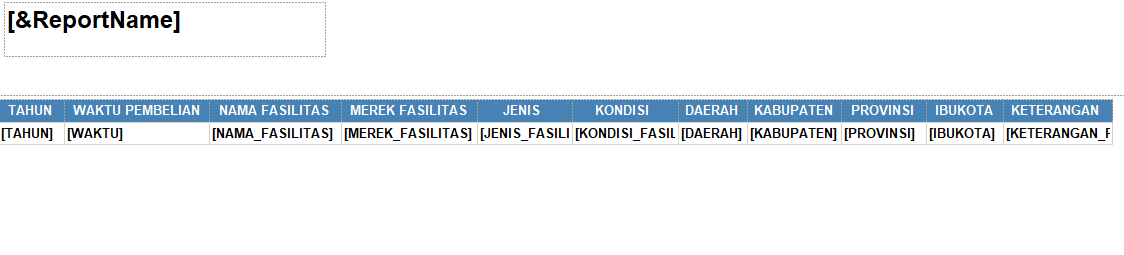
Pada menu *toolbax* terdapat menu *pilihan report item* yang terdiri dari menu pointer, textbox, line, table, matrix, rectangle, list, image, subreport, chart, gauge, map, data bar, sparkline, dan indicator, pilih *table* untuk membuat *report fasilitas* dalam bentuk tabel. Berikut dapat dilihat pada gambar 5.100 dibawah ini:



**Gambar 5.100 Tampilan *Report Items***

Pada gambar 5.100 merupakan tampilan dari *report items* yang ada pada SQL *Services Report*.

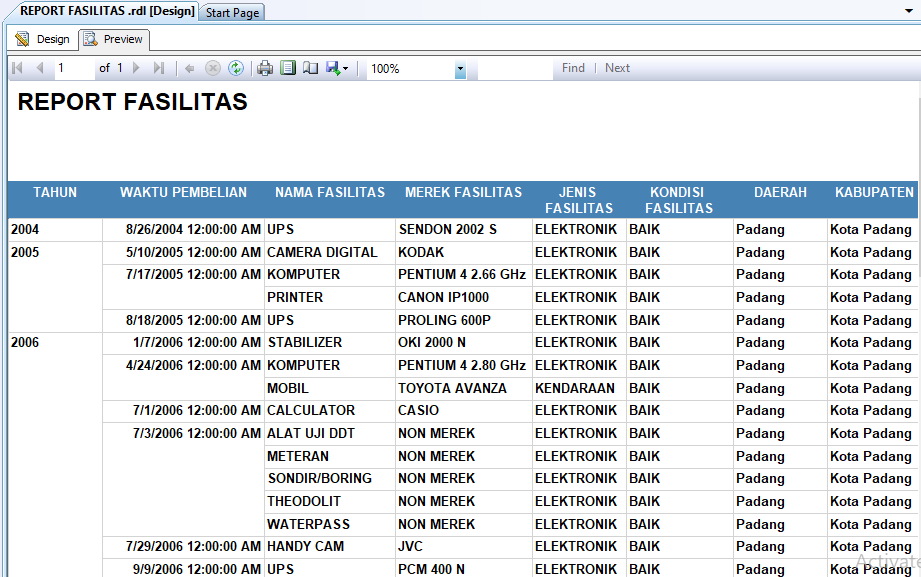
Setelah itu buat *design report* berdasarkan kebutuhan. *Design reporting* dari report fasilitas kemudian *preview* untuk melihat hasil dari tabel fasilitas yang telah dibuat. Berikut dapat dilihat pada gambar 5.101 dibawah ini:



**Gambar 5.101 Tampilan *Table Design* Fasilitas**

Pada gambar 5.101 merupakan tampilan design table pada fasilitas yang terdiri dari tahun, waktu pembelian, nama fasilitas, merek fasilitas, jenis fasilitas, kondisi fasilitas, daerah, kabupaten, provinsi, ibukota dan keterangan fasilitas.

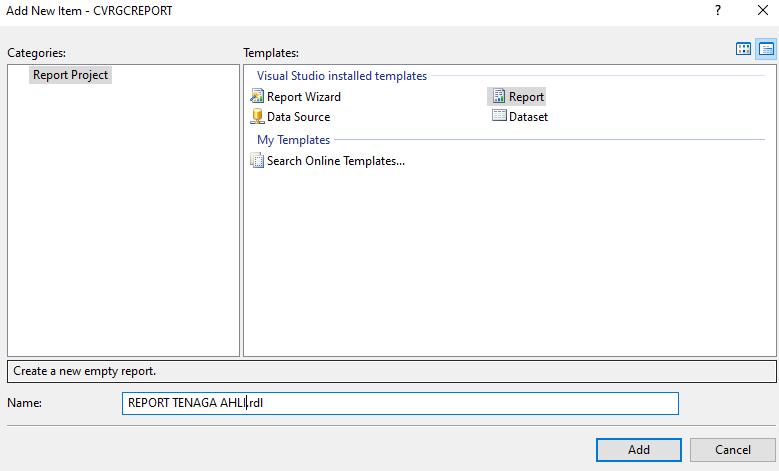
Klik *preview* untuk melihat hasil tabel fasilitas. Berikut dapat dilihat pada gambar 5.102 dibawah ini:



**Gambar 5.102 Tampilan *Table Preview* Fasilitas**

Pada gambar 5.102 merupakan tampilan hasil *tabel design* yang dibuat setelah dilakukan *review* pada *design* tersebut.

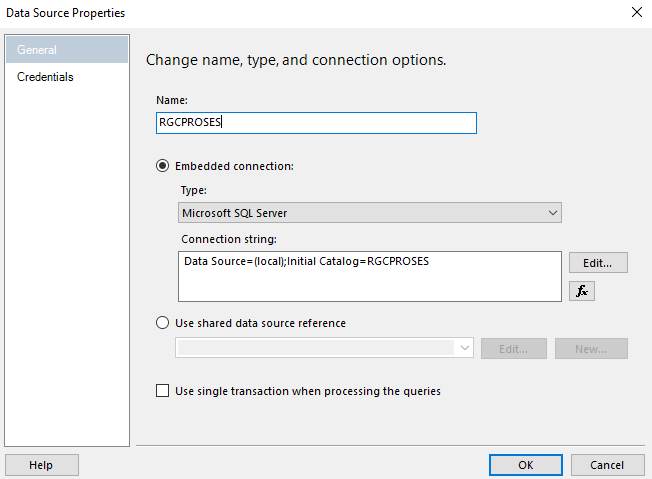
Untuk membuat report baru klik kanan pada *Shared Dataset*, kemudian klik *add* dan pilih *report.* Ganti nama untuk REPORT TENAGA AHLI yang akan di tampilkan. Klik *add* seperti gambar 5.103 dibawah ini.



**Gambar 5.103 Tampilan *Report Project* Tenaga Ahli**

Pada gambar 5.103 merupakan tampilan dari *properties report*. Buat nama berdasarkan laporan yang akan dibuat dan klik Add.

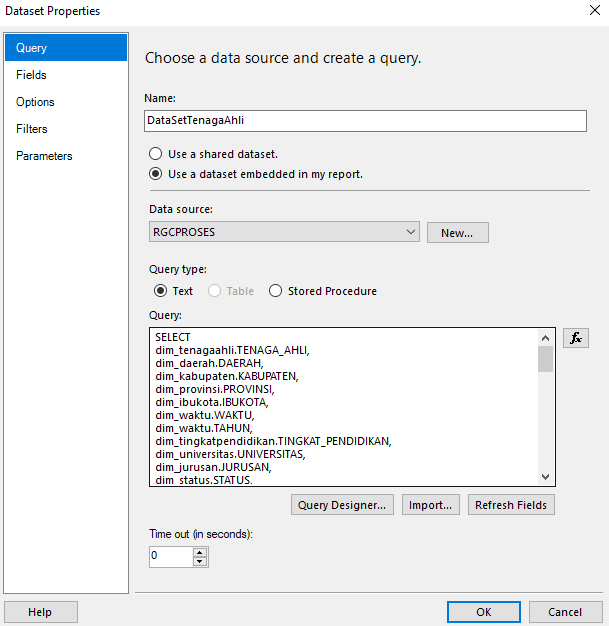
Setelah klik *add* akan muncul tampilan report data disamping kanan pada *project* yang akan dibuat. Klik *data source* sesuaikan data source project dengan *data source* yang ada pada reporting project dan klik OK. Berikut dapat dilihat pada gambar 5.104 dibawah ini:



**Gambar 5.104 Tampilan *Data Source Properties***

Pada gambar 5.104 terdapat tampilan *data source properties* untuk memilih penyimpanan data yang telah tersimpan. Data yang disimpan diolah di panggil untuk menampilkan laporan pada report.

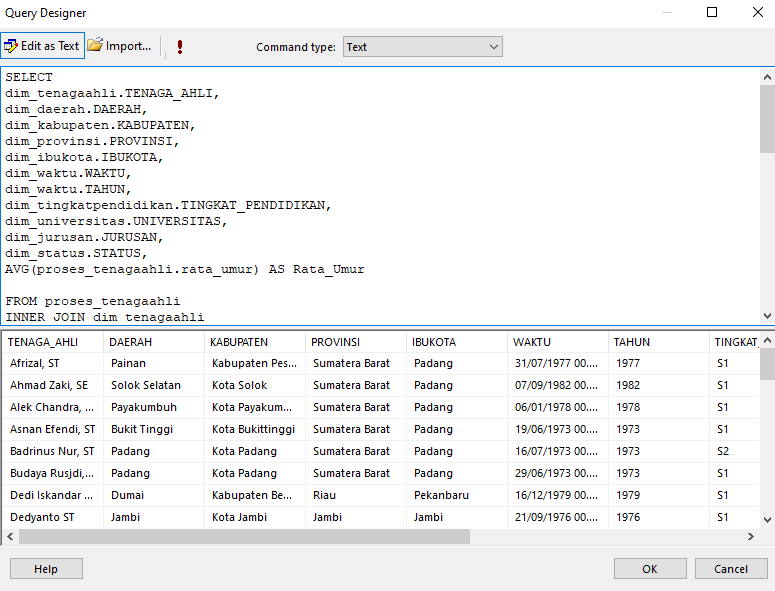
Kemudian klik kanan *dataset* dan pilih add *dataset*, ganti nama data set dengan nama report yang akan digunakan, pilih *use a data set embedded in my report*. Pilih *data source* yang telah dibuat sebelumnya dengan *query* type text. Pada *query* dan klik *query* designer. Berikut dapat dilihat pada gambar 5.105 dibawah ini:



**Gambar 5.105 Tampilan *Data Set Properties* Tenaga Ahli**

Pada gambar 5.105 tampilan *query* dari *dataset proeprties*. Ubah nama *dataset* menjadi datasetTenagaAhli, pilih data *source* RGCPROSES dimana data yang dibutuhkan tersimpan.

Setelah klik *query designer* buat *query* yang akan menampilkan tampilan report. Klik run (!) untuk menjalankan *query* setelah sukses klik OK. Berikut dapat dilihat pada gambar 5.106 dibawah ini:



**Gambar 5.106 Tampilan *Query Designer* Tenaga Ahli**

Pada gambar 5.106 proses untuk memasukan query pada laporan fakta tenaga ahli.

Untuk *Query* tenaga ahli :

**SELECT**

*dim\_tenagaahli.TENAGA\_AHLI,*

*dim\_daerah.DAERAH,*

*dim\_kabupaten.KABUPATEN,*

*dim\_provinsi.PROVINSI,*

*dim\_ibukota.IBUKOTA,*

*dim\_waktu.WAKTU,*

*dim\_waktu.TAHUN,*

*dim\_tingkatpendidikan.TINGKAT\_PENDIDIKAN,*

*dim\_universitas.UNIVERSITAS,*

*dim\_jurusan.JURUSAN,*

*dim\_status.STATUS,*

*AVG(proses\_tenagaahli.rata\_umur) AS Rata\_Umur*

FROM proses\_tenagaahli

*INNER JOIN dim\_tenagaahli*

*ON proses\_tenagaahli.TENAGA\_AHLI=dim\_tenagaahli.TENAGA\_AHLI*

*INNER JOIN dim\_daerah*

*ON proses\_tenagaahli.TEMPAT\_LAHIR=dim\_daerah.DAERAH*

*INNER JOIN dim\_kabupaten*

*ON dim\_daerah.kd\_kabupaten=dim\_kabupaten.kd\_kabupaten*

*INNER JOIN dim\_provinsi*

*ON dim\_kabupaten.kd\_provinsi=dim\_provinsi.kd\_provinsi*

*INNER JOIN dim\_ibukota*

*ON dim\_provinsi.kd\_ibukota=dim\_ibukota.kd\_ibukota*

*INNER JOIN dim\_waktu*

*ON proses\_tenagaahli.TANGGAL\_LAHIR=dim\_waktu.WAKTU*

*INNER JOIN dim\_tingkatpendidikan*

*ON proses\_tenagaahli.TINGKAT\_PENDIDIKAN=dim\_tingkatpendidikan.TINGKAT\_PENDIDIKAN*

*INNER JOIN dim\_universitas*

*ON proses\_tenagaahli.UNIVERSITAS=dim\_universitas.UNIVERSITAS*

*INNER JOIN dim\_jurusan*

*ON proses\_tenagaahli.JURUSAN=dim\_jurusan.JURUSAN*

*INNER JOIN dim\_status*

*ON proses\_tenagaahli.STATUS=dim\_status.STATUS*

**GROUP BY**

*dim\_tenagaahli.TENAGA\_AHLI,*

*dim\_daerah.DAERAH,*

*dim\_kabupaten.KABUPATEN,*

*dim\_provinsi.PROVINSI,*

*dim\_ibukota.IBUKOTA,*

*dim\_waktu.WAKTU,*

*dim\_waktu.TAHUN,*

*dim\_tingkatpendidikan.TINGKAT\_PENDIDIKAN,*

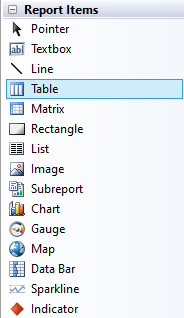
*dim\_universitas.UNIVERSITAS,*

*dim\_jurusan.JURUSAN,*

*dim\_status.STATUS,*

*proses\_tenagaahli.rata\_umur*

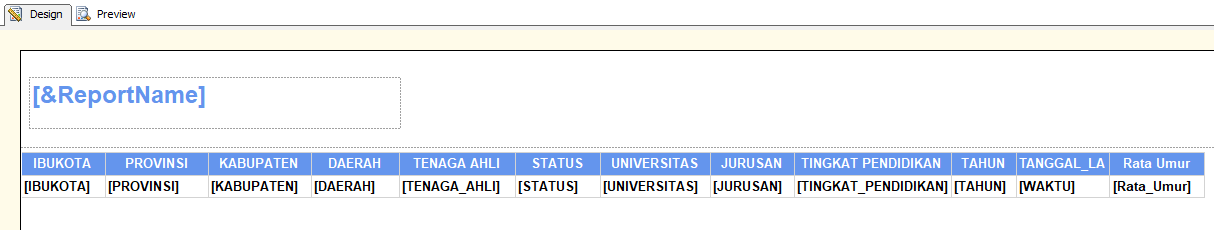
Pada menu *toolbax* terdapat menu *pilihan report items* dan pilih *table* untuk membuat report fasilitas dalam bentuk tabel. Berikut dapat dilihat pada gambar 5.107 dibawah ini:



**Gambar 5.107 Tampilan *Report Items***

Pada gambar 5.107 merupakan *report items* sebagai *design* *report* yang dibutuhkan untuk membuat laporan.

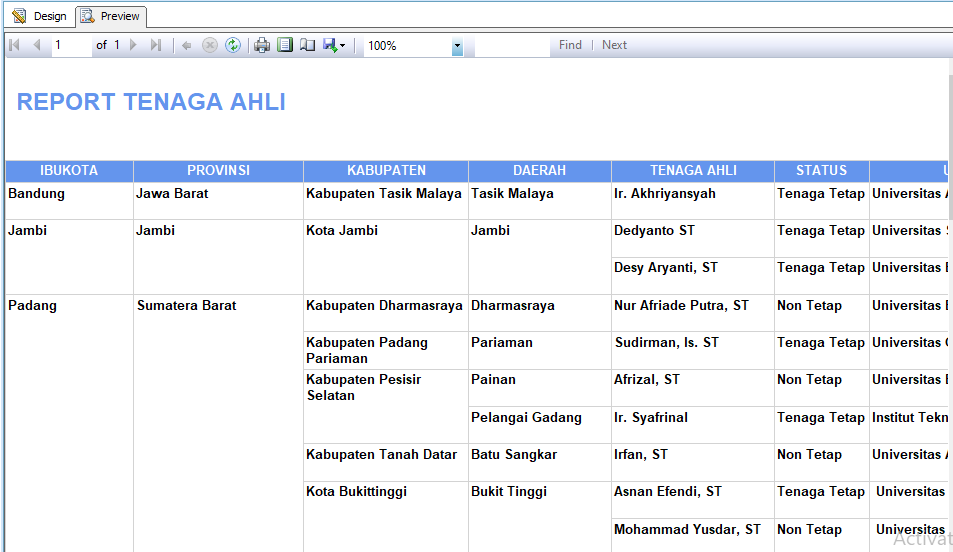
Setelah itu buat *design report* berdasarkan kebutuhan. Ini *design reporting* dari report tenaga ahli kemudian *preview* untuk lihat hasil dari tabel fasilitas yang telah dibuat. Berikut dapat dilihat pada gambar 5.108 dibawah ini:



**Gambar 5.108 Tampilan *Table Design* Tenaga Ahli**

Pada gambar 5.108 merupakan tampilan design table pada tenaga ahli yang terdiri dari ibukota, provinsi, kabupaten, daerah, tenaga ahli, status, universitas, jurusan, tingkat pendidikan, tahun lahir, tanggal lahir, dan rata-rata umur.

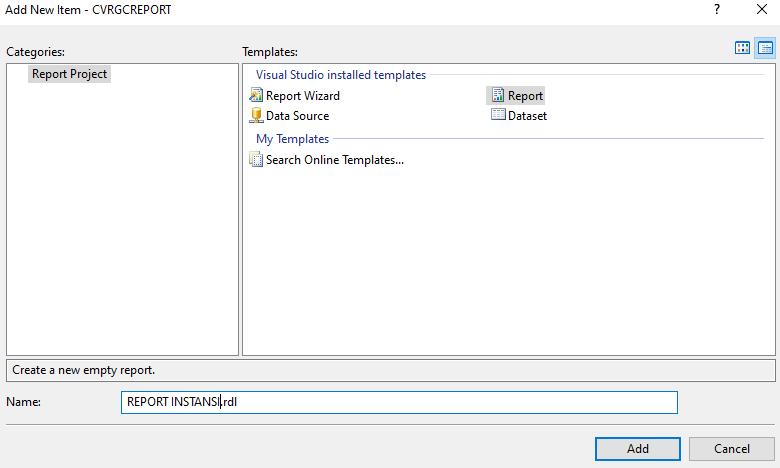
Klik *preview* untuk melihat hasil tabel tenaga ahli. Berikut dapat dilihat pada gambar 5.109 dibawah ini:



**Gambar 5.109 Tampilan *Table Preview* Tenaga Ahli**

Pada gambar 5.109 merupakan tampilan hasil *tabel design* yang dibuat setelah dilakukan *review* pada *design* tenaga ahli tersebut.

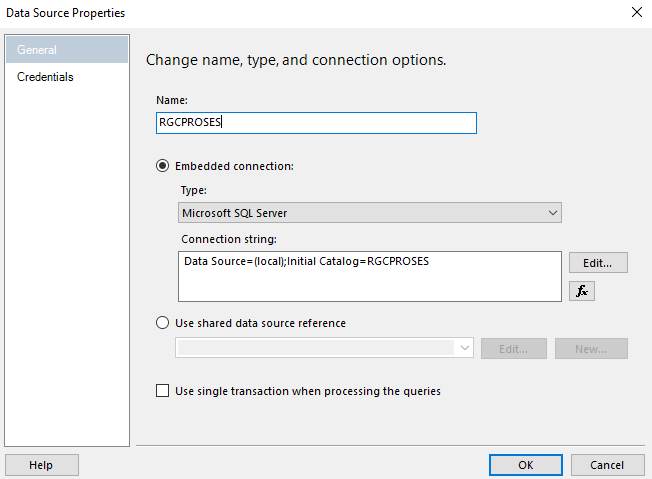
Untuk membuat report baru klik kanan pada *Shared Dataset*, kemudian klik *add* dan pilih report. Ganti nama untuk REPORT INSTANSI yang akan di tampilkan. Klik add seperti gambar 5.110 dibawah ini:



**Gambar 5.110 Tampilan *Report Project* Instansi**

Pada gambar 5.110 merupakan tampilan dari *properties report*. Buat nama berdasarkan laporan yang akan dibuat dan klik Add.

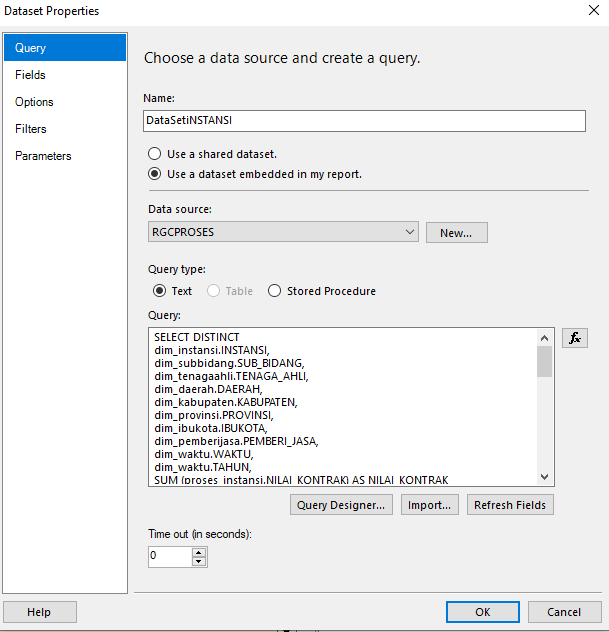
Setelah klik *add* akan muncul tampilan report data disamping kanan pada project yang akan dibuat. Klik *data source* sesuaikan *data source project* dengan *data source* yang ada pada *reporting project* dan klik OK. Berikut dapat dilihat pada gambar 5.111 dibawah ini:



**Gambar 5.111 Tampilan *Data Source Properties***

Pada gambar 5.111 terdapat tampilan *data source properties* untuk memilih penyimpanan data yang telah tersimpan. Data yang disimpan diolah di panggil untuk menampilkan laporan pada report.

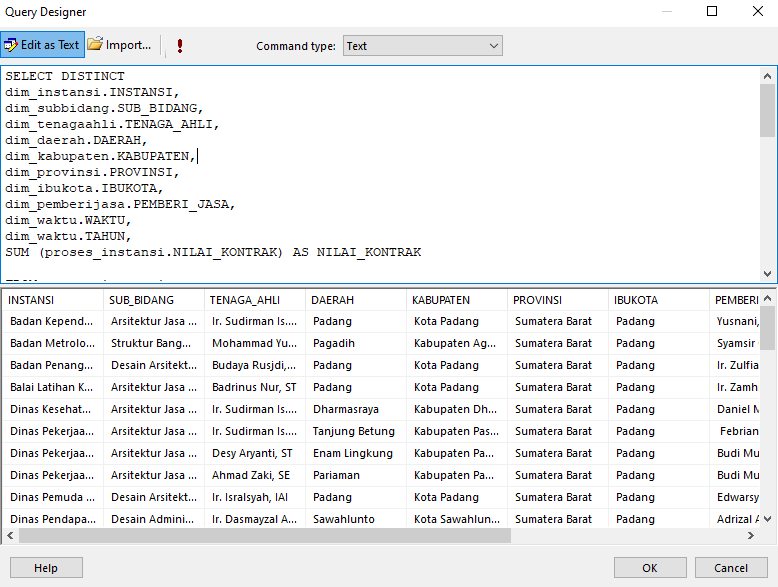
Kemudian klik kanan dataset dan pilih add *dataset*, ganti nama *dataset* pilih *use a data set embedded in my report*. Pilih data source dan klik *query designer*. Berikut dapat dilihat pada gambar 5.112 dibawah ini:



**Gambar 5.112 Tampilan *Data Set Properties* Instansi**

Pada gambar 5.112 tampilan *query* dari *dataset proeprties*. Ubah nama *dataset* menjadi datasetInstansi, pilih data *source* RGCPROSES dimana data yang dibutuhkan tersimpan.

Setelah klik *query designer*, buat *query* yang akan menampilkan tampilan report. Klik *run* (!) untuk menjalankan *query* yang telah dibuat klik OK. Berikut dapat dilihat pada gambar 5.113 dibawah ini:



**Gambar 5.113 Tampilan *Query Designer* Instansi**

Pada gambar 5.113 merupakan tampilan *edit text* untuk memasukan *query* untuk tahapan *design* laporan.

Untuk *Query* Instansi :

***SELECT DISTINCT***

*dim\_instansi.INSTANSI,*

*dim\_subbidang.SUB\_BIDANG,*

*dim\_tenagaahli.TENAGA\_AHLI,*

*dim\_daerah.DAERAH,*

*dim\_kabupaten.KABUPATEN,*

*dim\_provinsi.PROVINSI,*

*dim\_ibukota.IBUKOTA,*

*dim\_pemberijasa.PEMBERI\_JASA,*

*dim\_waktu.WAKTU,*

*dim\_waktu.TAHUN,*

*SUM (proses\_instansi.NILAI\_KONTRAK) AS NILAI\_KONTRAK*

***FROM proses\_instansi***

*INNER JOIN dim\_instansi*

*ON proses\_instansi.INSTANSI=dim\_instansi.INSTANSI*

*INNER JOIN dim\_subbidang*

*ON proses\_instansi.SUB\_BIDANG=dim\_subbidang.SUB\_BIDANG*

*INNER JOIN dim\_tenagaahli*

*ON proses\_instansi.TENAGA\_AHLI=dim\_tenagaahli.TENAGA\_AHLI*

*INNER JOIN dim\_daerah*

*ON proses\_instansi.LOKASI=dim\_daerah.DAERAH*

*INNER JOIN dim\_kabupaten*

*ON dim\_daerah.kd\_kabupaten=dim\_kabupaten.kd\_kabupaten*

*INNER JOIN dim\_provinsi*

*ON dim\_kabupaten.kd\_provinsi=dim\_provinsi.kd\_provinsi*

*INNER JOIN dim\_ibukota*

*ON dim\_provinsi.kd\_ibukota=dim\_ibukota.kd\_ibukota*

*INNER JOIN dim\_pemberijasa*

*ON proses\_instansi.PEMBERI\_JASA=dim\_pemberijasa.PEMBERI\_JASA*

*INNER JOIN dim\_waktu*

*ON proses\_instansi.KONTRAK\_INSTANSI=dim\_waktu.WAKTU*

***GROUP BY***

*dim\_instansi.INSTANSI,*

*dim\_subbidang.SUB\_BIDANG,*

*dim\_tenagaahli.TENAGA\_AHLI,*

*dim\_daerah.DAERAH,*

*dim\_kabupaten.KABUPATEN,*

*dim\_provinsi.PROVINSI,*

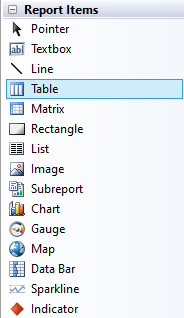
*dim\_ibukota.IBUKOTA,*

*dim\_pemberijasa.PEMBERI\_JASA,*

*dim\_waktu.WAKTU,*

*dim\_waktu.TAHUN*

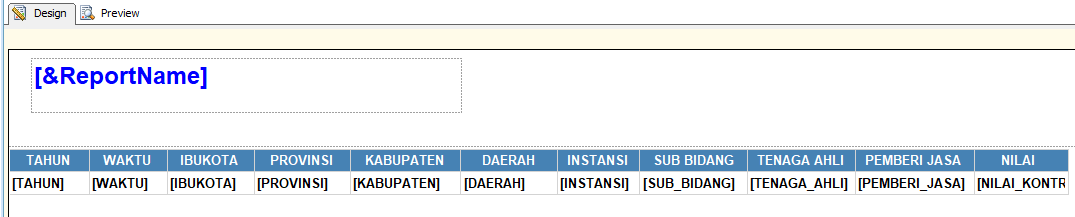
Pada menu toolbax terdapat menu *pilihan report items*, pilih table untuk membuat report fasilitas dalam bentuk tabel. Berikut dapat dilihat pada gambar 5.114 dibawah ini:



**Gambar 5.114 Tampilan *Report Items***

Pada gambar 5.114 merupakan *report items* sebagai design report yang dibutuhkan untuk membuat laporan.

Setelah itu buat *design report* berdasarkan kebutuhan. Berikut ini *design reporting* dari report tenaga ahli kemudian *preview* untuk lihat hasil dari tabel fasilitas yang telah dibuat. Berikut dapat dilihat pada gambar 5.115 dibawah ini:



**Gambar 5.115 Tampilan *Table Design* Instansi**

Pada gambar 5.115 merupakan tampilan design table pada instansi yang terdiri dari tahun, waktu, ibukota, provinsi, kabupaten, daerah, instansi, sub bidang, tenaga ahli, pemberi jasa, nilai kontrak.

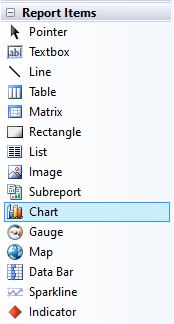
Klik *preview* untuk melihat hasil tabel tenaga ahli. Berikut dapat dilihat pada gambar 5.116 dibawah ini:



**Gambar 5.116 Tampilan *Table Review* Instansi**

Pada gambar 5.116 merupakan tampilan hasil *tabel design* yang dibuat setelah dilakukan *review* pada *design* tenaga ahli tersebut.

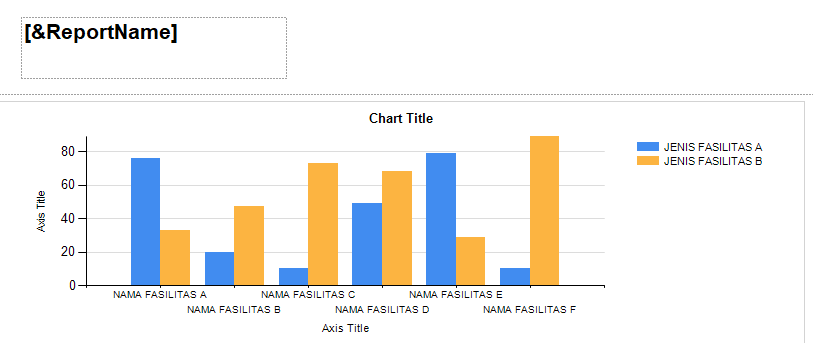
Report dapat dipilih berdasarkan kebutuhan. Report dibuat dalam bentuk grafik, pembuatan report grafik memiliki cara yang sama dengan cara report tabel. Perbedaan dari report tabel dan *chart t*erletak dalam pemilihan report yang dipilih berdasarkan kebutuhan. Ini merupakan *report chart* dari perusahaan CV. Restu Graha Cipta**.** *Select Chart* pada *report items* berikut dapat dilihat pada gambar 5.117 dibawah ini:



**Gambar 5.117 Tampilan *Report Items***

Pada gambar 5.117 merupakan tampilan dari *report items* yang ada pada SQL *Services Report*.

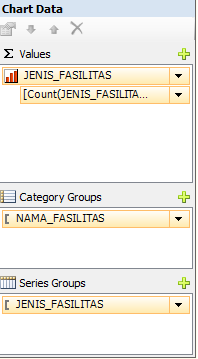
Setelah memilih *report item chart*, akan muncul tampilan grafik yang akan dibuat, query pada *data set* sesuaikan dengan tabel report fasilitas sebelumnya. Berikut dapat dilihat pada gambar 5.118 dibawah ini:



**Gambar 5.118 Tampilan *Design Chart* Fasilitas**

Pada gambar 5.118 merupakan tampilan design laporan dengan menggunakan report items chart. Langkah awal yang dilakukan dalam pembuatan laporan dengan menggunakan chart sama dengan menggunakan table, perbedaan dari laporan table dan chart hanya tampilan yang akan dalam proses pembuatan laporan.

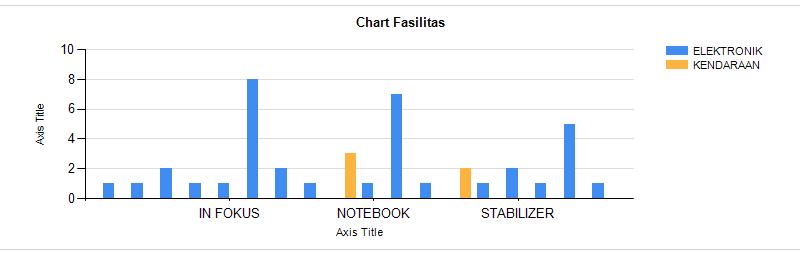
Pada tampilan grafik terdapat *chart data* untuk memasukan dimensi-dimensi yang telah dibuat pada query untuk membuat grafik berdasarkan kebutuhan. Berikut dapat dilihat pada gambar 5.119 dibawah ini:



**Gambar 5.119 Tampilan *Chart Data* Fasilitas**

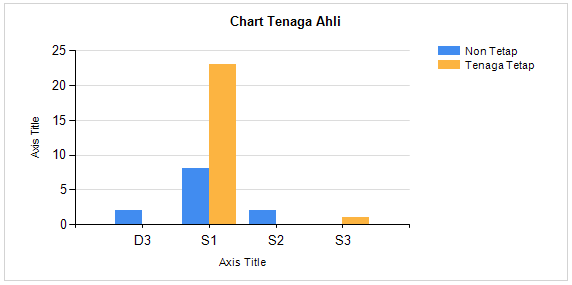
Pada gambar 5.119 merupakan tampilan dari chart data atau axes table untuk mengatur tampilan judul x dan y dalam chart yang terdiri dari *category groups* dan *series group*.

Langkah selanjutnya *review* *chart* untuk melihat hasil dari grafik yang telah kita buat. Berikut dapat dilihat pada gambar dibawah ini:



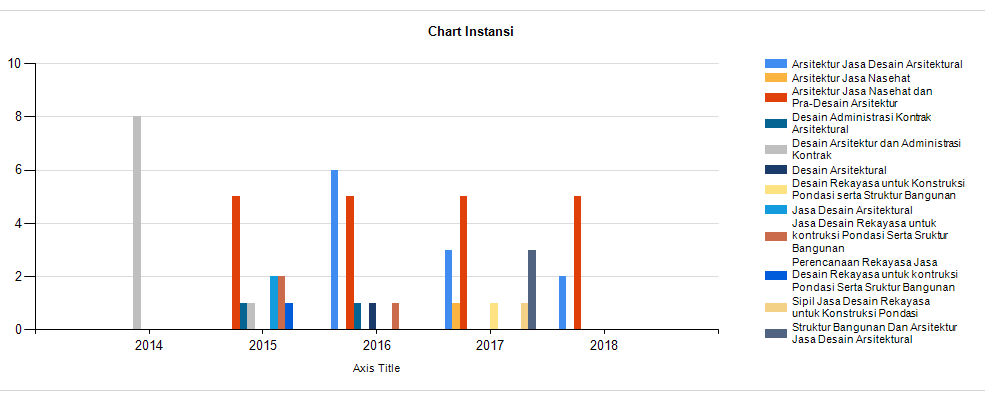
**Gambar 5.119 Tampilan *Chart Review* Fasilitas**

Pada gambar 5.119 merupakan contoh *design chart* fasilitas yang telah di *review*.



**Gambar 5.120 Tampilan *Chart Review* Tenaga Ahli**

Pada gambar 5.120 merupakan contoh *design chart* tenaga ahli yang telah di *review*.



**Gambar 5.121 Tampilan *Chart Review* Instansi**

Pada gambar 5.121 merupakan contoh *design chart* Instansi yang telah di *review*.

# 5.2 Pengujian

Pengujian merupakan proses yang digunakan untuk membantu dalam mengidentifikasi kebenaran, kelengkapan dan kualitas perangkat lunak yang dikembangkan. Pengujian pada penelitian ini menggunakan pengujian black box yaitu pengujian yang memastikan setiap bagian sesuai dengan alur dari proses yang telah ditetapkan dan memastikan semua kesalahan dapat ditangani oleh sistem.

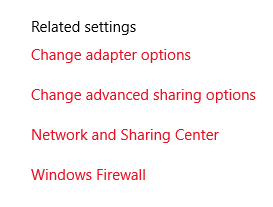
# Pengujian LAN

Aplikasi *data warehouse* dirancang akan dijalankan secara *Local Area Network* (LAN), pengujian ini bertujuan agar komputer server perusahaan dengan komputer pemimpin saling terhubung dan dapat berkomunikasi terutama dalam proses pengiriman data. Artinya *admin* perusahaan bertindak sebagai *client* pada saat tertentu dapat melakukan akses data pada komputer pemimpin yang bersangkutan. Sedangkan komputer pemimpin bertindak sebagai *server* pada saat tertentu juga dapat menyediakan data yang dibutuhkan oleh *admin*.

Adapun langkah-langkah dalam pengujian LAN dapat diraikan sebagai berikut :

1. Siapkan dua komputer maupun laptop agar dapat melakukan pengujian.
2. Siapkan kabel *Cross Rj-45* yang berfungsi sebagai alat penghubung antara komputer dan *switch/hub*.
3. Lakukan Setting IP pada komputer *server*, dengan langkah-langkah sebagai berikut :

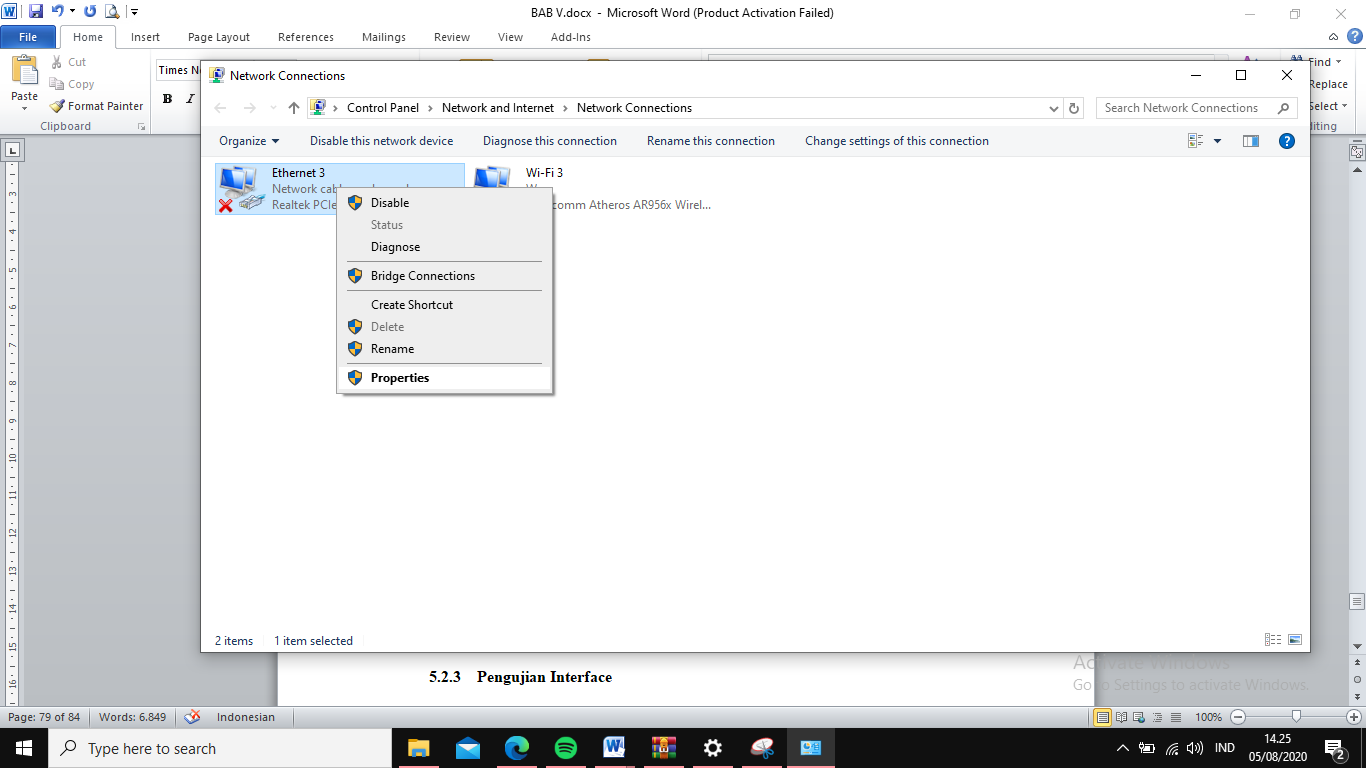
*Open Network and Sharing Center* klik menu *change adapter settings* seperti gambar 5.122 dibawah sebagai berikut :



**Gambar 5.122 Change Adapter options**

Pada gambar 5.122 merupakan *related setting* untuk melakukan settingan yang akan dilakukan.

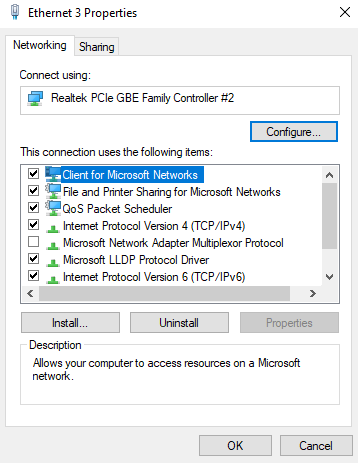
Pilih *Local Data Connection* dan klik kanan koneksi tersebut kemudian klik properties seperti gambar 5.123 dibawah ini :



**Gambar 5.123 *Properties***

Pada gambar 5.123 *properties options* pada *etherne*t jika melakukan pemilihan dengan klik kanan. Ethernet merupakan jejaring komputer untuk jaringan area lokal (LAN) yang nantinya akan menghubungkan antaraclient dan server.

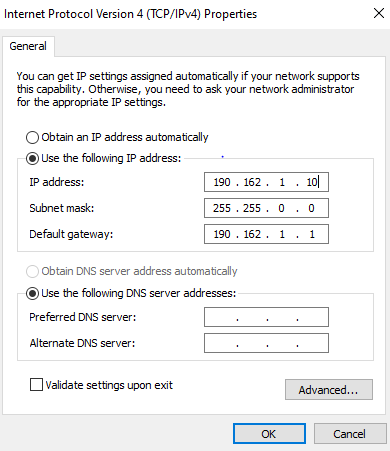
Langkah selanjutnya akan tampil halaman *Local Area Connection* yang dapat dilihat pada gambar 5.124 berikut ini :



**Gambar 5.124 Internet Protocol Version 4 (TCP/Ipv4)**

Pada gambar 5.124 diatas menunjukan halaman *Local Area Connection* yang kemudian pilih *Internet Protocol Version 4 (TCP/Ipv4)*.

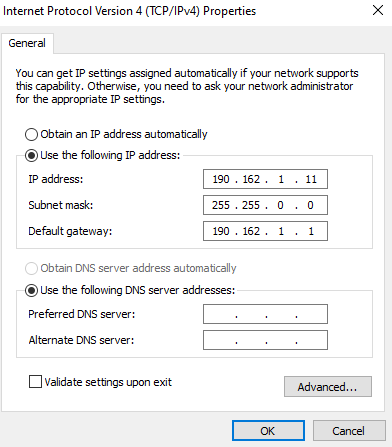
*Setting IP Address Server* adalah *setting IP Address* untuk *server* yang dapat dilihat pada gambar 5.125 berikut ini :



**Gambar 5.125 IP Setting Server**

Pada gambar 5.125 untuk mengatur *IP* pada komputer *server* dengan memilih “*Use The Following IP Address*” dan atur *IP address*, *subnet mask* dan *default gateway* pada form yang tersedia.

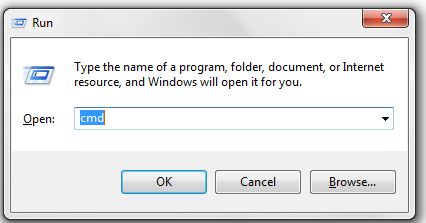
Langkah selanjutnya adalah setting *IP* pada komputer yang akan dijadikan *client*. Untuk langkah mengatur IP pada komputer client sama dengan cara sebelumnya hanya yang berbeda yaitu *IP Address* yang akan diisi pada komputer *client*, jika komputer *server IP address* beralamat 190.162.1.10, maka pada komputer *client* buat dengan alamat IP yang berbeda agar tidak terjadi kesalahan dalam menghubungkan dan pengujian nantinya seperti gambar 5.126 berikut ini :



**Gambar 5.126 IP Setting Client**

Pada gambar 5.126 diatas menunjukan pengisian *IP address* dengan IP 190.162.1.111 di komputer client dengan *subnet mask* 255.255.0.0 dan *default gateway* 190.162.1.1.

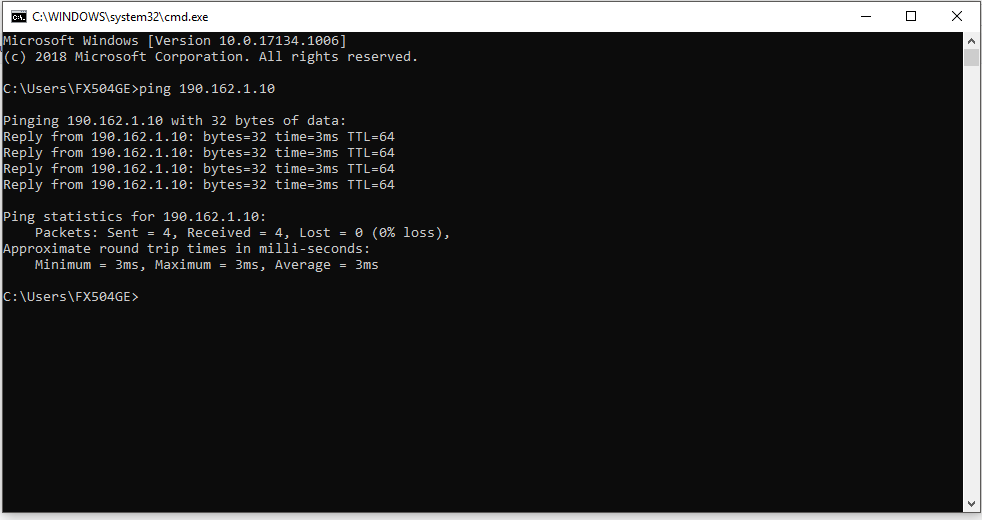
Masuk CMD Untuk langkah selanjutnya masuk kedalam CMD pada komputer *client*, yang dapat dilihat pada gambar 5.127 berikut ini :



**Gambar 5.127 Start CMD**

Pada gambar 5.127 melakukan input dan pengaturan pada kedua komputer *client* maupun *server* lanjutkan dengan menguji apakah kedua komputer sudah terhubung atau belum. Untuk komputer *server* caranya pada *keyboard* tekan bersamaan antara *WIN+R* maka akan muncul layar *Run* pada komputer, lalu ketikkan “*cmd*” dan klik OK. CMD digunakan untuk melakukan ping melihat troubleshooting pada jaringan komputer.

Masukan *IP address* di komputer *Client* dengan ketik *IP address server* pada cmd yang ada pada komputer *client* dapat dilihat pada gambar 5.128 berikut ini :



**Gambar 5.128 Ping Pada Komputer Client**

Pada gambar 5.128 merupakan halaman *Command Prompt* untuk *server* ketik alamat IP yang ada pada komputer *client* “ping 190.162.1.10” lalu tekan *enter*, maka akan muncul tampilan bahwa antara komputer *server* dengan *client* sudah terhubung.

Setelah melakukan pengujian maka kedua komputer baik *server* maupun *client* sudah dapat melakukan komunikasi maupun lakukan *sharing* file data nantinya.

# Pengujian Aplikasi

Pengujian aplikasi merupakan pengujian untuk mengetahui kesalahan dalam struktur data atau melakukan akses database secara eksternal. Melihat kesalahan kinerja serta fungsi-fungsi yang ada pada aplikasi. Pegujian ini bertujuan untuk menunjukkan aplikasi yang telah dirancang sudah berjalan dengan benar atau belum, hal ini bertujuan agar aplikasi dapat memberikan informasi dalam pembuatan sistem yang belum berjalan dengan sempurna, dan pengujian ini dapat menjelaskan kegunaan aplikasi yang telah dirancang.

Pengujian yang dilakukan dapat dilihat dalam keterangan tabel 5.1 penggunaan sistem sebagai berikut :

**Tabel 5.1 Tabel Pengujian Aplikasi**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **NO** | **PENGUJIAN** | **HASIL YANG DIHARAPKAN** | **HASIL PENGUJIAN** | **KESIMPULAN** |
| 1 | Login | Menampilkan halaman login diawal aplikasi data warehouse | Berjalan sesuai yang diharapkan | Berhasil |
| 2 | Menampilkan Halaman Dashboard | Menampilkan informasi dalam bentuk grafis, yang merupakan tampilan tools untuk visualisasi dalam tampilan informasi | Berjalan sesuai yang diharapkan | Berhasil |
| 3 | Menampilkan Halaman Profile Perusahaan | Menampilkan profile perusahaan | Berjalan sesuai yang diharapkan | Berhasil |
| 4 | Menampilkan Halaman Tabel Laporan perusahaan | Menampilkan data-data perusahaan secara struktur yang dibuat dalam bentuk tabel untuk dapat dilihat dan dapat membantu dalam memberikan informasi. | Berjalan sesuai yang diharapkan | Berhasil |
| 5 | Menampilkan Grafik | Menampilkan data yang dalam bentuk angka secara detail dan menjelaskan objek yang saling berhubungan secara singkat dan jelas. | Berjalan sesuai yang diharapkan | Berhasil |
| 6 | Logout | Berfungsi untuk keluar dari aplikasi dan akan kembali ke halaman login | Berjalan sesuai harapan | Berhasil |

Halaman Login merupakan tampilan login yang dapat dilihat dari gambar 5.129 berikut :



**Gambar 5.129 Halaman Login**

Pada gambar 5.129 merupakan tampilan halaman login. Pemimpin dan admin harus menginputkan username dan password dengan benar pada form yang tersedia untuk dapat melanjutkan aktifitas yang terdapat didalam aplikasi.

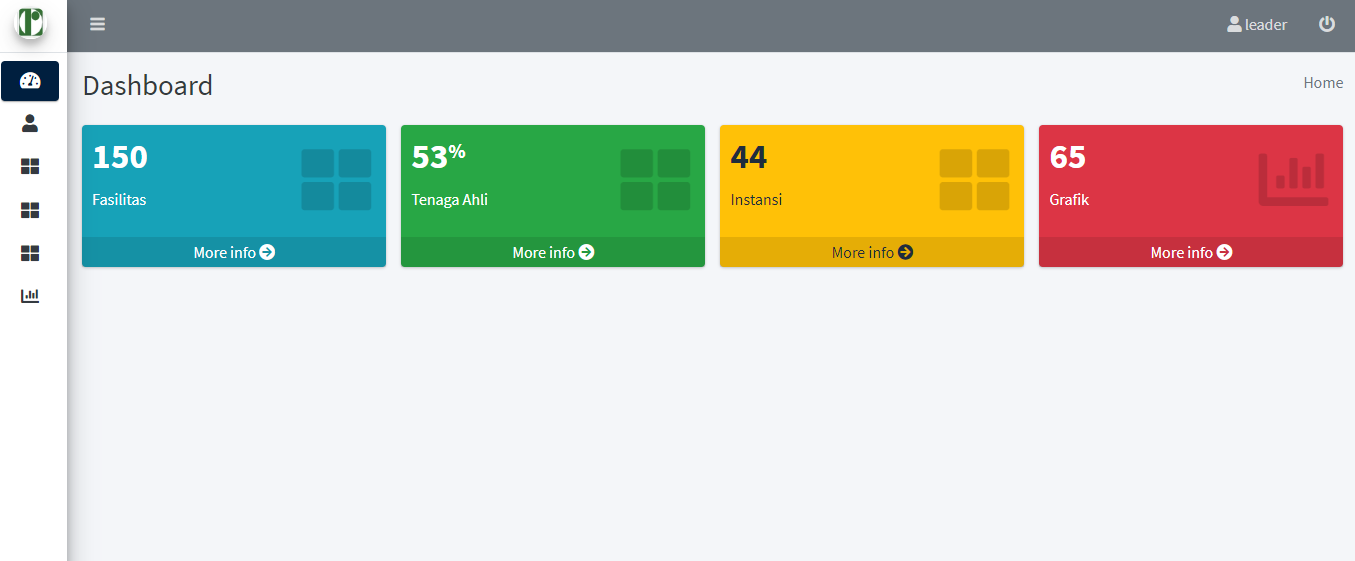
Halaman Profile merupakan halaman profile dari perusahaan yang dapat dilihat dari gambar 5.130 dibawah ini :



**Gambar 5.130 Halaman Profile**

Pada gambar 5.130 merupakan tampilan halaman profile pada pengujian inetrface. Pada halaman profile merupakan halaman yang berisi profile perusahaan Restu Graha Cipta. Di dalam profile perusahaan terdiri dari N.P.W.P perusahaan, alamat perusahaan, kota/kabupaten, provinsi, kode pos, negara, nomor hp, fax dan email dari perusahaan. Terdapat kata pengantar tujuan dari perusahaan.

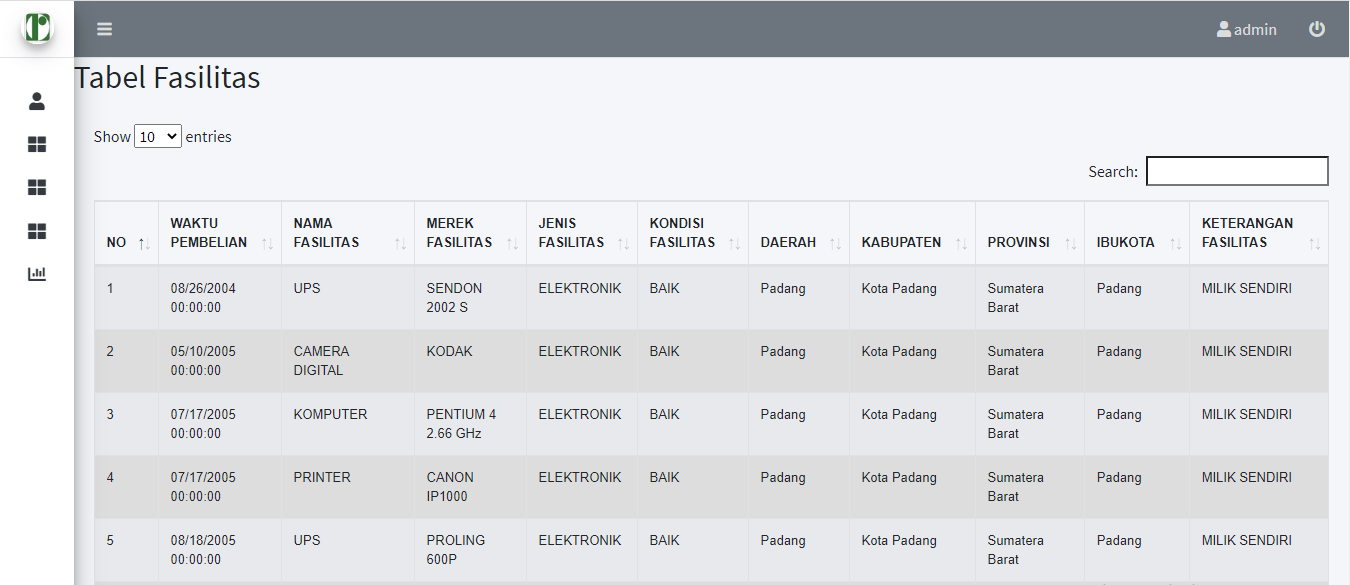
Halaman Dashboard merupakan halaman pusat control panel platform untuk mengatur seluruh kegiatan di website yang dapat dilihat pada gambar 5.131 dibawah ini:



**Gambar 5.131 Halaman Halaman Dashboard**

Pada gambar 5.131 merupakan halaman dashboard sebagai alat untuk mengontrol semua kegiatan di website.

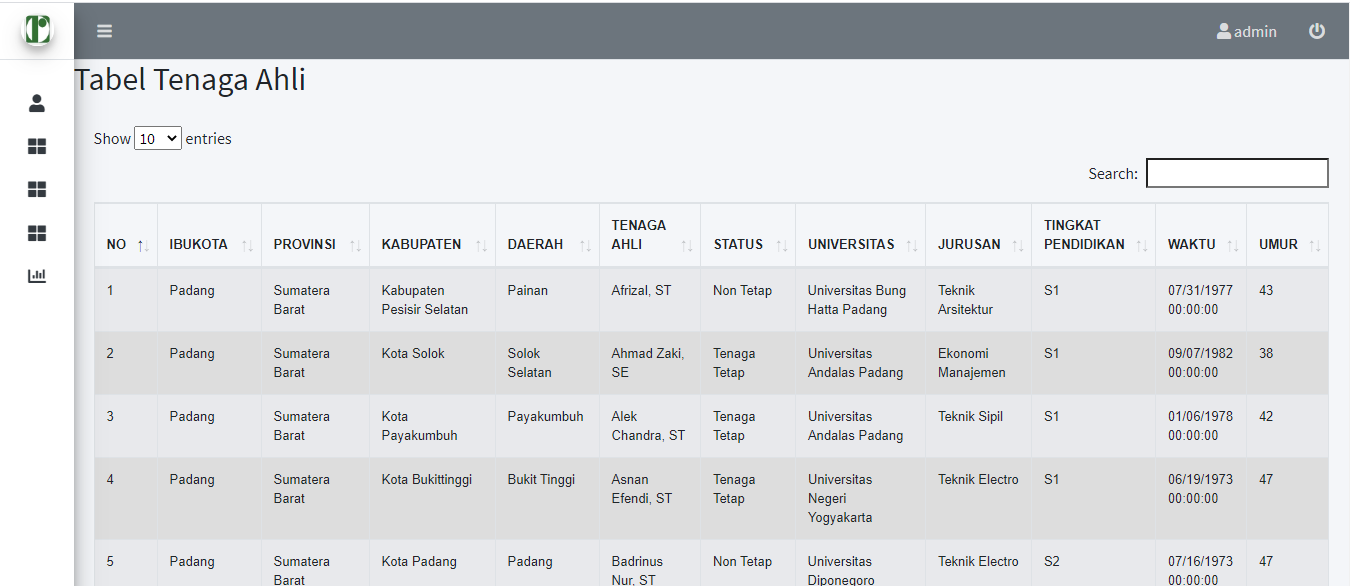
Halaman tabel merupakan halaman data pada perusahaan pada interface pengujian terdiri dari 3 jenis yaitu fasilitas, tenaga ahli dan instansi. Berikut dapat dilihat pada gambar 5.132 tabel fasilitas pada pengujian interface dibawah ini:



**Gambar 5.132 Halaman Tabel Fasilitas**

Pada gambar 5.132 merupakan tampilan dari tabel tenaga fasilitas yang ada pada pengujian interface. Tabel tenaga ahli terdiri dari waktu pembelian, nama fasilitas, merek fasilitas, jenis fasilitas, kondisi fasilitas, daerah, kabupaten, provinsi, ibu kota dan keterangan fasilitas.

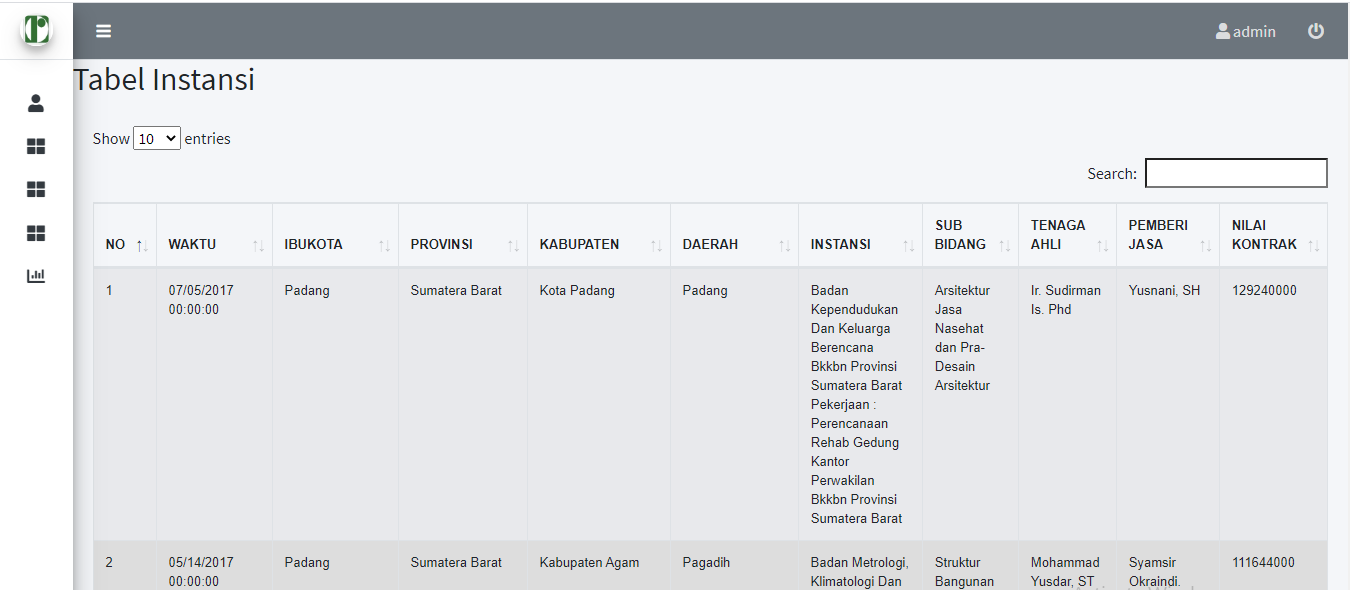
Berikut dapat dilihat gambar 5.133 merupakan tabel tenaga ahli pada pengujian aplikasi dibawah ini:



**Gambar 5.133 Halaman Tabel Tenaga Ahli**

Pada gambar 5.133 merupakan tampilan dari tabel tenaga ahli yang ada pada pengujian interface. Tabel tenaga ahli terdiri dari ibu kota, provinsi, kabupaten, daerah, tenaga ahli, status, universitas, jurusan, tngkat pendidikan, waktu dan umur tenaga ahli.

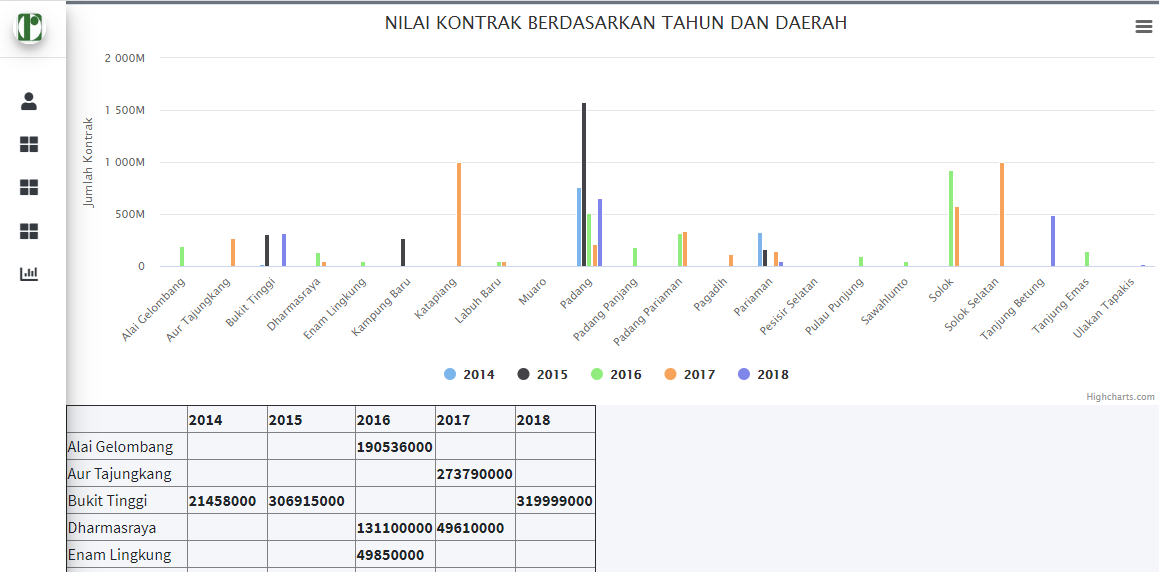
Berikut dapat dilihat gambar 5.134 tabel instansi pada pengujian interface dibawah ini:



**Gambar 5.134 Halaman Tabel Instansi**

Pada gambar 5.134 merupakan tampilan dari tabel instansi yang ada pada pengujian interface. Tabel waktu, ibu kota, provinsi, kabupaten, daerah, instansi, sub bidang, tenaga ahli, pemberi jasa dan nilai kontrak.

Halaman Grafik Data merupakan halaman untuk menampilkan grafik atau *chart* yang telah dibuat berdasarkan kebutuhan dari perusahaan. Berikut dapat dilihat pada gambar 5.135 tampilan grafik pada pengujian interface dibawah ini:



**Gambar 5.135 Halaman Tampilan Grafik**

Pada gambar 5.135 merupakan tampilan dari grafik yang akan ditampilkan pada pengujian interface. Pada grafik dapat melihat data yang diolah dalam bentuk chart berdasarkan kebutuhan. Untuk melihat perkembangan data berdasarkan angka secara detail dan melihat perbandingan data yang saling berhubungan secara singkat dan jelas.

# DAFTAR PUSTAKA

[1] S. M. N. Huda And J. Sutrisno, “Analisa Perancangan Data Warehouse Dan Aplikasi Online Analytical Processing Pengajuan Kredit Pada Pt Bfi Finance Indonesia Tbk,” *Idealis Indones. J. Inf. Syst.*, Vol. 1, No. 3, Pp. 354–359, 2018.

[2] A. R. Simon And T. C. Hammergren, *Data Warehousing For Dummies*. John Wiley & Sons, 2009.

[3] I. P. A. E. Pratama, *Handbook Data Warehouse : Teori Dan Praktik Berbasiskan Open Source*, Informatik. Bandung, 2018.

[4] T. Connolly And C. Begg, *Database System : A Practical Approach To Design, Implementation, And Management. Fourth Edition*, Fourth Edi. England, 2005.

[5] K. Khotimah And Sriyanto, “Perancangan Dan Implementasi Data Warehouse Untuk Mendukung Sistem Akademik (Studi Kasus Pada Stkip Muhammadiyah Kotabumi),” *J. Teknol. Inf. Magister*, Vol. 2, No. 01, Pp. 94–107, 2017.

[6] A. Supriyatna, “Sistem Analisis Data Mahasiswa Menggunakan Aplikasi Online Analytical Processing (Olap) Data Warehouse,” *Pilar Nusa Mandiri*, Vol. 12, No. 1, Pp. 62–71, 2016.

[7] M. K. A. Ghani, M. M. Jaber, And N. Suryana, “Telemedicine Supported By Data Warehouse Architecture,” *Arpn J. Eng. Appl. Sci*, Vol. 10, No. 2, Pp. 415–417, 2015.

[8] R. Permana, “Perancangan Data Mart Bagian Penjualan Motor Bekas ( Used Motor Cycle ) Pada Cv . Atlas Motor,” Vol. 3, No. 1, 2013.

[9] Z. Nur And I. Mukhlash, “Implementasi Business Intelligence Pada Manajemen Report Bank Xyz,” *J. Sains Dan Senni Pomits*, Vol. 3, No. Bisnis Intelijen, Pp. 16–21, 2014.

[10] D. Marlius, “Loyalitas Nasabah Bank Nagari Syariah Cabang Bukittinggi Dilihat Dari Kualitas Pelayanan,” *J. Pundi*, Vol. 1, No. 3, Pp. 237–250, 2018, Doi: 10.31575/Jp.V1i3.60.