

MODUL DATA WAREHOUSE & BUSINESS INTELLIGENCE

UNIVERSITAS TEKNOLOGI MATARAM

Pengertian Data, Informasi dan Database

Sebelum kita membahas tentang data warehouse, hal yang harus dipahami terlebih dahulu yaitu pengertian tentang data, informasi dan database. Menurut Steven Alter, data merupakan fakta,gambar atau suara yang mungkin atau tidak berhubungan atau berguna bagi tugas tertentu.
Menurut McLeod, data terdiri dari fakta-fakta dan angka yang secara relatif tidak berarti bagi pemakai. Sedangkan informasi adalah data yang sudah diproses atau data yang memiliki arti.
Disini kita dapat melihat bahwa data merupakan "*suatu bentuk keterangan-keterangan yang belum diolah atau dimanipulasi sehingga belum begitu berarti bagi sebagian pemakai. Sedangkan informasi merupakan data yang sudah di olah sehingga memiliki arti*".
Menurut James A. O'Brien Database adalah suatu koleksi terintegrasi dimana secara logika berhubungan dengan record dari file.
Menurut Fatansyah, Database adalah kumpulan data yang saling berhubungan yang disimpan secara bersama sedemikian rupa dan tanpa pengulangan(redudansi) yang tidak perlu, untuk memenuhi berbagai kebutuhan.

Jadi Database adalah tempat penyimpanan data yang saling berhubungan secara logika, sehingga bisa digunakan untuk mendapatkan suatu informasi yang diperlukan oleh suatu organisasi atau perusahaan. Sedangkan data yang diperoleh suatu organisasi atau perusahaan umumnya didapat dari kegiatan operasional sehari-hari atau hasil dari transaksi. Dari perkembangan model database, muncullah apa yang disebut dengan data warehouse.

I.2. Pengertian Data Warehouse

Pengertian Data Warehouse dapat bermacam-macam namun mempunyai inti yang sama, seperti pendapat beberapa ahli berikut ini :

Menurut W.H. Inmon dan Richard D.H., *data warehouse* adalah koleksi data yang mempunyai sifat berorientasi subjek,terintegrasi,time-variant, dan bersifat tetap dari koleksi data dalam mendukung proses pengambilan keputusan management.

Menurut Vidette Poe, *data warehouse* merupakan database yang bersifat analisis dan *read only* yang digunakan sebagai fondasi dari sistem penunjang keputusan.

Menurut Paul Lane, *data warehouse* merupakan database relasional yang didesain lebih kepada *query* dan analisa dari pada proses transaksi, biasanya mengandung *history* data dari proses transaksi dan bisa juga data dari sumber lainnya. *Data warehouse* memisahkan beban

kerja analisis dari beban kerja transaksi dan memungkinkan organisasi menggabung/konsolidasi data dari berbagai macam sumber.

Jadi, *data warehouse* merupakan metode dalam perancangan database, yang menunjang DSS (Decission Support System) dan EIS (Executive Information System). Secara fisik *data warehouse* adalah database, tapi perancangan data warehouse dan database sangat berbeda. Dalam perancangan database tradisional menggunakan normalisasi, sedangkan pada *data warehouse* normalisasi bukanlah cara yang terbaik.

Dari definisi-definisi yang dijelaskan tadi, dapat disimpulkan *data warehouse* adalah database yang saling bereaksi yang dapat digunakan untuk *query* dan analisisis, bersifat orientasi subjek, terintegrasi, time-variant,tidak berubah yang digunakan untuk membantu para pengambil keputusan.

I.3. Istilah-istilah yang berhubungan dengan data warehouse

Istilah-istilah yang berkaitan dengan data warehouse :

1. Data Mart

   Adalah suatu bagian pada data warehouse yang mendukung pembuatan laporan dan analisa data pada suatu unit, bagian atau operasi pada suatu perusahaan.

2. On-Line Analytical Processing(OLAP)

   Merupakan suatu pemrosesan database yang menggunakan tabel fakta dan dimensi untuk dapat menampilkan berbagai macam bentuk laporan, analisis, query dari data yang berukuran besar.

3. On-Line Transaction Processing(OLTP)

   Merupakan suatu pemrosesan yang menyimpan data mengenai kegiatan operasional transaksi sehari-hari.

4. Dimension Table

Tabel yang berisikan kategori dengan ringkasan data detail yang dapat dilaporkan. Seperti laporan laba pada tabel fakta dapat dilaporkan sebagai dimensi waktu(yang berupa perbulan, perkwartal dan pertahun).

5. Fact Table

   Merupakan tabel yang umumnya mengandung angka dan data history dimana *key* (kunci) yang dihasilkan sangat unik, karena *key* tersebut terdiri dari *foreign key*(kunci asing) yang merupakan *primary key* (kunci utama) dari beberapa dimension table yang berhubungan.

6. DSS

   Merupkan sistem yang menyediakan informasi kepada pengguna yang menjelaskan bagaimana sistem ini dapat menganalisa situasi dan mendukung suatu keputusan yang baik.

I.4. KaKrakteristik *Data Warehouse*

Karakteristik data warehouse menurut Inmon, yaitu :

1. ***Subject Oriented*** **(Berorientasi subject)**

Data warehouse berorientasi subject artinya data warehouse didesain untuk menganalisa data berdasarkan subject-subject tertentu dalam organisasi,bukan pada proses atau fungsi aplikasi tertentu.

Data warehouse diorganisasikan disekitar subjek-subjek utama dari perusahaan(customers,products dan sales) dan tidak diorganisasikan pada area-area aplikasi utama(customer invoicing,stock control dan product sales). Hal ini dikarenakan kebutuhan dari data warehouse untuk menyimpan data-data yang bersifat sebagai penunjang suatu keputusan, dari pada aplikasi yang berorientasi terhadap data.

Jadi dengan kata lain, data yang disimpan adalah berorientasi kepada subjek bukan terhadap proses. Secara garis besar perbedaan antara data operasional dan data warehouse yaitu :

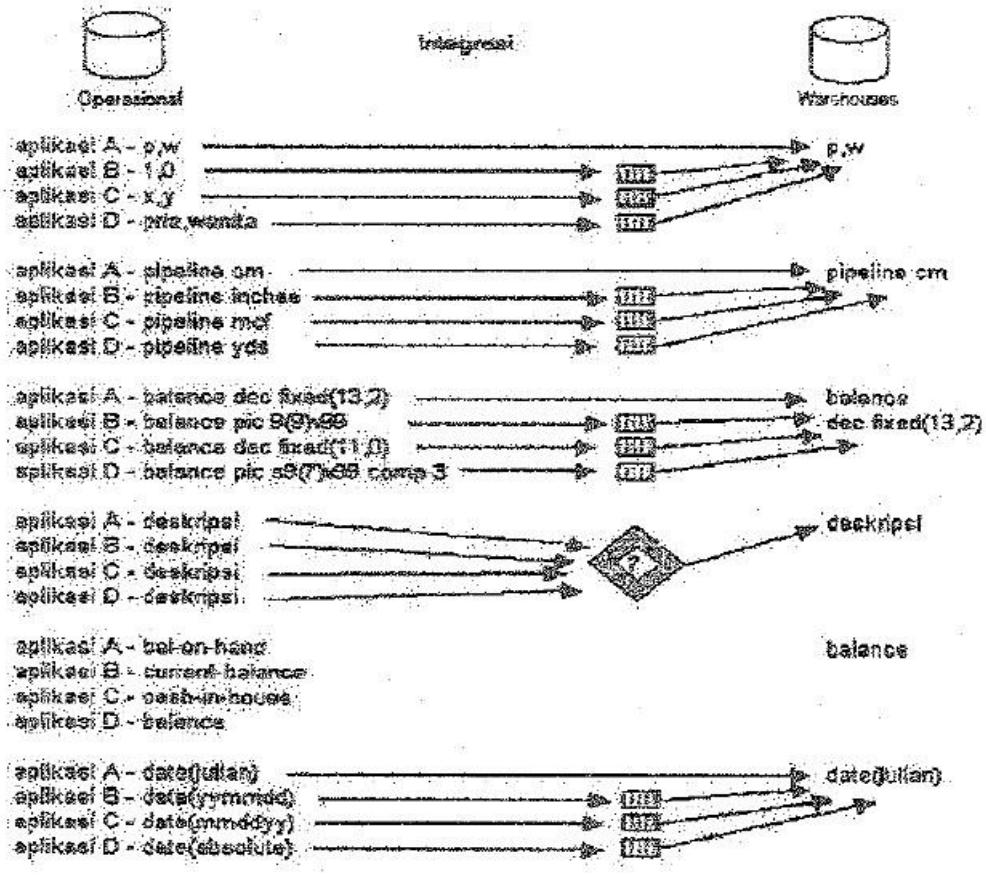
Data Operasional	Data Warehouse
Dirancang berorientasi hanya pada aplikasi dan fungsi tertentu	Dirancang berdasar pada subjek-subjek tertentu(utama)
Focusnya pada desain database dan proses	Focusnya pada pemodelan data dan desain data
Berisi rincian atau detail data	Berisi data-data history yang akan dipakai dalam proses analisis
Relasi antar table berdasar aturan terkini(selalu mengikuti rule(aturan) terbaru)	Banyak aturan bisnis dapat tersaji antara tabel-tabel

2. *Integrated* (Terintegrasi)

Data Warehouse dapat menyimpan data-data yang berasal dari sumber-sumber yang terpisah kedalam suatu format yang konsisten dan saling terintegrasi satu dengan lainnya. Dengan demikian data tidak bisa dipecah-pecah karena data yang ada merupakan suatu kesatuan yang menunjang keseluruhan konsep data warehouse itu sendiri.

Syarat integrasi sumber data dapat dipenuhi dengan berbagai cara sepeti konsisten dalam penamaan variable,konsisten dalam ukuran variable,konsisten dalam struktur pengkodean dan konsisten dalam atribut fisik dari data.

Contoh pada lingkungan operasional terdapat berbagai macam aplikasi yang mungkin pula dibuat oleh developer yang berbeda. Oleh karena itu, mungkin dalam aplikasi-aplikasi tersebut ada variable yang memiliki maksud yang sama tetapi nama dan format nya berbeda. Variable tersebut harus dikonversi menjadi nama yang sama dan format yang disepakati bersama. Dengan demikian tidak ada lagi kerancuan karena perbedaan nama, format dan lain sebagainya. Barulah data tersebut bisa dikategorikan sebagai data yang terintegrasi karena kekonsistenannya.



Integrasi Data
Warehouse

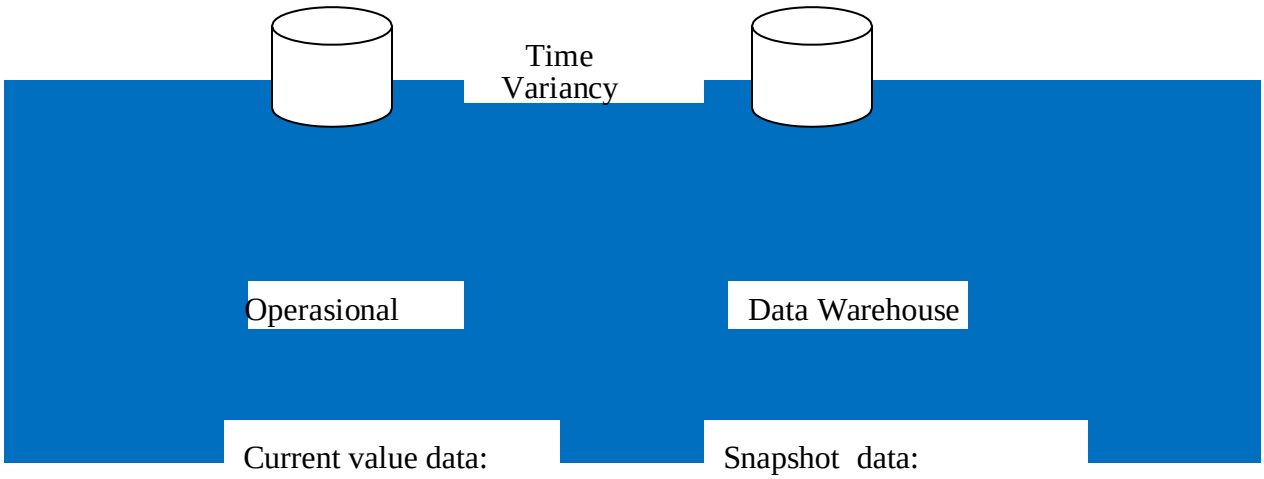
3. *Time-variant (*Rentang Waktu*)*

Seluruh data pada data warehouse dapat dikatakan akurat atau valid pada rentang waktu tertentu. Untuk melihat interval waktu yang digunakan dalam mengukur keakuratan suatu data warehouse, kita dapat menggunakan cara antara lain :

- Cara yang paling sederhana adalah menyajikan data warehouse pada rentang waktu tertentu, misalnya antara 5 sampai 10 tahun ke depan.
- Cara yang kedua, dengan menggunakan variasi/perbedaan waktu yang disajikan dalam data warehouse baik implicit maupun explicit secara explicit dengan unsur waktu dalam hari, minggu, bulan dsb. Secara implicit misalnya pada saat data

tersebut diduplikasi pada setiap akhir bulan, atau per tiga bulan. Unsur waktu akan tetap ada secara implisit didalam data tersebut.

- Cara yang ketiga,variasi waktu yang disajikan data warehouse melalui serangkaian snapshot yang panjang. Snapshot merupakan tampilan dari sebagian data tertentu sesuai keinginan pemakai dari keseluruhan data yang ada bersifat read-only.



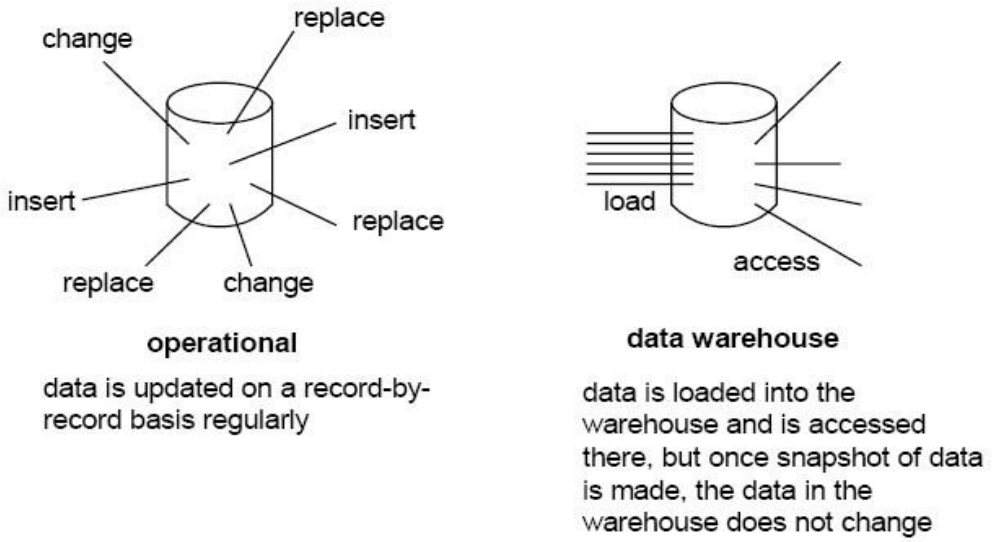
Time Variance Data Warehouse

4. Non-Volatile

Karakteristik keempat dari data warehouse adalah non-volatile,maksudnya data pada data warehouse tidak di-*update* secara *real time* tetapi di *refresh* dari sistem operasional secara

reguler. Data yang baru selalu ditambahkan sebagai suplemen bagi database itu sendiri dari pada sebagai sebuah perubahan. Database tersebut secara kontinyu menyerap data baru ini, kemudian secara incremental disatukan dengan data sebelumnya.

Berbeda dengan database operasional yang dapat melakukan update,insert dan delete terhadap data yang mengubah isi dari database sedangkan pada data warehouse hanya ada dua kegiatan memanipulasi data yaitu loading data (mengambil data) dan akses data (mengakses data warehouse seperti melakukan query atau menampilan laporan yang dibutuhkan, tidak ada kegiatan updating data).



Non Volatile Data Warehouse

Perlunya Data Warehouse

Seperti pengertian-pengertian yang kita sebutkan sebelumnya, data warehouse diperlukan bagi para pengambil keputusan manajemen dari suatu organisasi/perusahaan.

Dengan adanya data warehouse, akan mempermudah pembuatan aplikasi-aplikasi DSS dan EIS karena memang kegunaan dari data warehouse adalah khusus untuk membuat suatu database yang dapat digunakan untuk mendukung proses analisa bagi para pengambil keputusan.

Tugas-tugas Data warehouse

Ada empat tugas yang bisa dilakukan dengan adanya data warehouse Menurut Williams, keempat tugas tersebut yaitu:

a. Pembuatan laporan

Pembuatan laporan merupakan salah satu kegunaan data warehouse yang paling umum dilakukan. Dengan menggunakan query sederhana didapatkan laporan perhari,perbulan, pertahun atau jangka waktu kapanpun yang diinginkan.

b. On-Line Analytical Processing (OLAP)

Dengan adanya data warehouse,semua informasi baik detail maupun hasil summary yang dibutuhkan dalam proses analisa mudah didapat.

OLAP mendayagunakan konsep data multi dimensi dan memungkinkan para pemakai menganalisa data sampai mendetail, tanpa mengetikkan satupun perintah SQL. Hal ini dimungkinkan karena pada konsep multi dimensi, maka data yang berupa fakta yang sama bisa dilihat dengan menggunakan fungsi yang berbeda. Fasilitas lain yang ada pada sofware OLAP adalah fasilitas rool-up dan drill-down. Drill-down adalah kemampuan untuk melihat detail dari suatu informasi dan roll-up adalah kebalikannya.

c. Data mining

Data mining merupakan proses untuk menggali(mining) pengetahuan dan informasi baru dari data yang berjumlah banyak pada data warehouse, dengan menggunakan kecerdasan buatan (Artificial Intelegence), statistik dan matematika. Data mining merupakan teknologi yang diharapkan dapat menjembatani komunikasi antara data dan pemakainya.

Beberapa solusi yang diberikan data mining antara lain :

1. Menebak target pasar

   Data mining dapat mengelompokkan (clustering) model-model pembeli dan melakukan klasifikasi terhadap setiap pembeli dan melakukan klasifikasi terhadap setiap pemebeli sesuai dengan karakteristik yang diinginkan.
2. Melihat pola beli dari waktu ke waktu

   Data mining dapat digunakan untuk melihat pola beli dari waktu ke waktu.
3. *cross-market analysis*

   Data mining dapat dimanfaatkan untuk melihat hubungan antara satu produk dengan produk lainnya.
4. Profil pelanggan

   Data mining bisa membantu pengguna untuk melihat profil pembeli sehingga dapat diketahui kelompok pembeli tertentu cenderung kepada suatu produk apa saja.
5. Informasi *summary*

   Data mining dapat membuat laporan summary yang bersifat multi dimensi dan dilengkapi dengan informasi statistik lainnya.

d. Proses informasi executive

Data warehouse dapat membuat ringkasan informasi yang penting dengan tujuan membuat keputusan bisnis, tanpa harus menjelajahi keseluruhan data. Dengan menggunakan data warehouse segala laporan telah diringkas dan dapat pula mengetahui segala rinciannya secara lengkap, sehingga mempermudah proses pengambilan keputusan. Informasi dan data pada laporan data warehouse menjadi target informative bagi user.

Keuntungan Data Warehouse

Data warehouse merupakan pendekatan untuk menyimpan data dimana sumber-sumber data yang heterogen(yang biasanya tersebar pada beberapa database OLTP) dimigrasikan untuk penyimpanan data yang homogen dan terpisah. Keuntungan yang didapatkan dengan menggunakan data warehouse tersebut dibawah ini (Ramelho).

- Data diorganisir dengan baik untuk *query* analisis dan sebagai bahan untuk pemrosesan transaksi.
- Perbedaan diantara struktur data yang heterogen pada beberapa sumber yang terpisah dapat diatasi.
- Aturan untuk transformasi data diterapkan untuk memvalidasi dan mengkonsolidasi data apabila data dipindahkan dari database OLTP ke *data warehouse*.
- Masalah keamanan dan kinerja bisa dipecahkan tanpa perlu mengubah sistem produksi.

Membangun data warehouse tentu saja memberikan keuntungan lebih bagi suatu perusahaan, karena data warehouse dapat memberikan keuntungan strategis pada perusahaan tersebut melebihi pesaing-pesaing mereka. Keuntungan tersebut diperoleh dari beberapa sumber (Sean Nolan,Tom Huguelet):

- Kemampuan untuk mengakses data yang besar
- Kemampuan untuk memiliki data yang konsistent
- Kemampuan kinerja analisa yang cepat
- Mengetahui adanya hasil yang berulang-ulang
- Menemukan adanya celah pada *business knowledge* atau *business process.*

- Mengurangi biaya administrasi
- Memberi wewenang pada semua anggota dari perusaahan dengan menyediakan kepada mereka informasi yang dibutuhkan agar kinerja bisa lebih efektif.

1. Sebuah Pendekatan Praktis

Untuk dapat merumuskan sebuah pendekatan dalam organisasi anda, anda perlu memeriksa apa yang organisasi anda inginkan. Apakah organisasi anda mencari hasil jangka panjang atau data mart yang cepat hanya untuk beberapa subjek untuk saat ini? Apakah organisasi anda menginginkan implementasi yang cepat, konsep yang terbukti? Atau organisasi anda membutuhkan beberapa pendekatan praktis lainnya?

Walaupun kedua pendekatan pada modul sebelumnya *bottom-up* dan *top-down* masing-masing memiliki keuntungannya sendiri, Pendekatan keduanya dapat dikompromikan menjadi suatu pendekatan yang praktis. Pengusung pendekatan praktis adalah Ralph Kimball, seorang penulis dan ahli Data Warehouse.Tahap-tahap dalam pendekatan praktis ini adalah sebagai berikut :

1. Merencanakan dan mendefiniskan kebutuhkan pada semua level korporat.
2. Membuat arsitektur menyeluruh untuk Data Warehouse yang komplit
3. Konformasi dan standarisasi isi data
4. Mengimplementasikan Data Warehouse sebagai seperangkat supermart, satu kali pada satu waktu

Pada pendekatan praktis ini, anda harus masuk ke hal paling mendasar dan menentukan apa yang benar-benar dibutuhkan organisasi anda dalam jangka panjang. Kunci pendekatan ini adalah pertama kali merencanakan level enterprise. Kemudian menentukan isi data untuk masing-masing supermart. Supermart harus benar-benar menggambarkan data mart. Anda buat supermart ini satu kali. Sebelum implementasi, anda harus memastikan bahwa isi data diantara beberapa supermart dikonformasikan dengan tipe datanya, panjang karakter, presisi dan semantik. Sebuah Data Warehouse oleh karenanya juga dikatakan sebagai gabungan (union) dari semua data mart yang telah dikonformasi. Data Mart tunggal atau individual ditargetkan untuk kelompok bisnis tertentu dalam skala enterprise, namun kumpulan semua data mart membentuk keseluruhan yang terintegrasi yang disebut *Enterprise Data Warehouse*

2. Komponen-komponen Data Warehouse

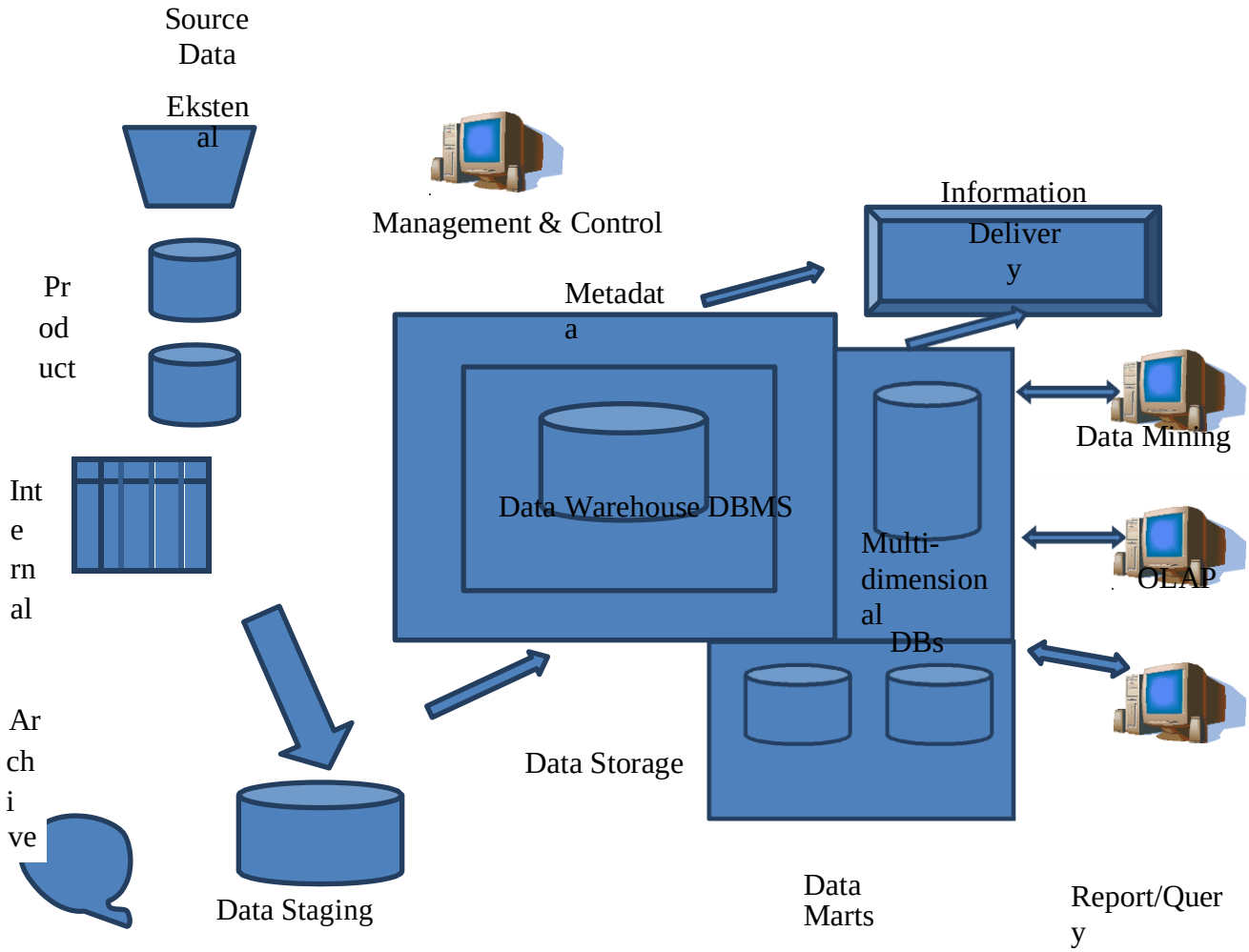
Ketika membangun sebuah sistem operasional seperti entry pesanan, proses klaim, atau tabungan, kita memasukkan beberapa komponen untuk membentuk sistem. Komponen *front-end* terdiri dari GUI (*Graphical User Interface*) untuk menghubungkan dengan

Pengguna bagi proses input data. Komponen penyimpanan data memasukkan sistem manajemen database seperti Oracle, informix atau Microsoft SQL Server. Komponen display adalah tampilan di layar dan laporan bagi pengguna. Interface data dan perangkat lunak jaringan membentuk komponen jaringan. Bergantung pada kebutuhan informasi dan *framework* organisasi kita menyusun komponen-komponen ini dengan seoptimal mungkin.

Arsitektur adalah susunan komponen yang sesuai. Anda membangun sebuah Data Warehouse dengan komponen perangkat keras dan perangkat lunak. Untuk memenuhi kebutuhan organisasi anda, anda menyusun *building block* ini yang memaksimumkan keuntungan.

Gambar 2.1. menunjukkan komponen dasar Data Warehouse. Pada gambar komponen *source data* ditunjukkan disebelah kiri, kemudian pada bagian *Building Block* berikutnya komponen *Data Staging*. Di tengah, terdapat komponen *Data Storage* yang mengatur data Data Warehouse. Komponen ini tidak hanya menyimpan dan mengatur data, tapi juga menjaga *track* data dengan menggunakan tempat penyimpanan data. Komponen *Information delivery*ditampilkan pada sebelah kanan terdiri dari seluruh cara-cara yang berbeda untuk menjadikan informasi Data Warehouse tersedia bagi pengguna.

Walaupun anda membangun Data Warehouse bagi perusahaan besar, sebuah grosir utama dengan rantai penjualan dan toko yang tersedia di banyak negara atau lembaga perbankan global, komponen dasarnya sama. Masing-masing Data Warehouse diletakkan bersama dengan *building block* yang sama. Perbedaan utamanya pada masing-masing organisasi adalah bagaimana *building block* ini disusun. Variasinya adalah dimana beberapa blok dibuat lebih dominan dibandingkan lainnya dalam arsitektur.



Gambar 2.1. Komponen atau *Building Block* Data Warehouse

2.1. Komponen *Data Source*

Komponen *Data Source* dibagi menjadi 4 kategori utama, yakni :

a. Data Produksi (*Production Data*)

Kategori data ini berasal dari beberapa sistem operasional enterprise. Berdasarkan kebutuhan informasi dalam Data Warehouse, anda memilih segmen data dari sistem operasional yang berbeda. Ketika berurusan dengan data ini, anda akan mendapatkan banyak variasi format data. Hal lain yang perlu diperhatikan adalah platform perangkat keras yang berbeda.Data juga didukung sistem operasi dan sistem database yang berbeda.

Pada sistem operasional, kueri informasi bersifat sempit. Anda melakukan kueri sistem operasional mengenai informasi mengenai instan spesifik dari suatu objek bisnis. Misalkan anda membutuhkan untuk hanya mengetahui nama dan alamat pelanggan tunggal, atau melihat sebuah invoice dan tagihan item-item pada invoice tersebut.

Dalam sistem operasional anda tidak melakukan kueri yang luas. Seluruh kueri dapat diprediksi, karena tidak membutuhkan kueri yang berjalan pada sistem operasional yang berbeda. Karakteristik yang pasti dan cukup merepotkan pada data produksi adalah perbedaan data. Tantangan anda adalah menstandarisasi dan mentransformasi data yang berbeda dari berjenis sistem produksi, mengkonversi data, dan mengintegrasikan bagian-bagiannya menjadi data yang berguna bagi penyimpanan dalam Data Warehouse.

b. Data Internal (*Internal Data)*

Dalam setiap organisasi, pengguna memiliki lembar kerja yang bersifat "rahasia" begitu juga dokumen, profil pelanggan dan juga database per departemen. Ini adalah data internal, bagian yang boleh jadi berguna dalam Data Warehouse.

Jika organisasi melakukan bisnis ke pelanggan yang bersifat satu-ke-satu dan kontribusi masing-masing pelanggan bersifat berarti. Profil pelanggan individual menjadi sangat penting untuk dipertimbangkan.

Anda tidak dapat mengabaikan data internal yang disimpan dalam file rahasia dalam organisasi anda. Departemen IT harus bekerja dengan departemen *user* untuk mengumpulkan data internal.

Internal Data menambah komplesitas ke dalam proses transformasi dan integrasi data sebelum disimpan ke dalam Data Warehouse. Anda harus menentukan strategi pengumpulan data dari lembar kerja, menemukan cara mengambil data dari dokumen tekstual dan mengaitkannya menjadi database per departemen untuk memperoleh data yang berhubungan dari sumber-sumber tersebut.

c. Data Arsip (*Archived Data*)

Sistem operasional ditujukan untuk menjalankan bisnis saat ini. Dalam setiap sistem operasional, anda secara periodik mengambil data lama dan

menyimpannya dalam file arsip. Sikon organisasi yang menentukan seberapa sering dan seberapa banyak data operasional yang diarsipkan untuk disimpan. Beberapa data diarsipkan setelah satu tahun. Terkadang data ditinggalkan dalam sistem database operasional selama 5 tahun.

Ada banyak metode pengarsipan yang berbeda yang ada hingga saat ini. Ada metode pengarsipan bertingkat, pada tingkat pertama data yang paling akhir diarsipkan ke arsip database yang terpisah yang mungkin masih online. Pada tingkat kedua,data yang lebih lama diarsipkan kebentuk flat file pada media penyimpanan berupa disk. Pada tahapan berikutnya data yang paling lama diarsipkan ke media penyimpanan data atau microfilm.

Organisasi pasti membutuhkan data historis untuk analisis setiap waktu. Untuk memperoleh informasi historis, dapat dipeoleh dari data set yang diarsipkan. Tergantung pada kebutuhan Data Warehouse , organisasi harus memasukkan data historis yang mencukupi. Tipe data ini berguna dalam melihat pattem atau analisis trend.

d. Data Eksternal (*External Data*)

Kebanyakan eksektif bergantung pada data dari sumber eksternal untuk persentase informasi tinggi yang dibutuhkan. Mereka menggunakan statistik untuk industri yang dihasilkan agen eskternal. Mereka juga menggunakan *share data* pesaing. Dan juga indikator keuangan bagi bisnis mereka untuk mengukur kinerja mereka.

Contoh Data Warehouse rental mobil berisikan data jadwal produksi saat ini dari perusahaan automobil ternama. Data Eksternal di dalam Data Warehouse membantu perusahaan rental mobil merencanakan manajemen armada mereka.

Tujuan yang dilayani sumber data eksternal tidak dapat dipenuhi ketersediaan data didalam organisasi. Umumnya data yang bersumber dari luar tidak sesuai dengan format yang ada di organisasi. Karena itu dibutuhkan konversi data menjadi format internal dan juga tipe datanya. Anda harus melakukan pengorganisasian transmisi data dari sumber eksternal. Beberapa sumber menyediakan informasi reguler maupun dengan interval.

2.2. Komponen *Data Staging*

Setelah mengekstrak data dari beberapa sistem operasional dan sumber eksternal, kemudian data disiapkan untuk disimpan dalam Data Warehouse. Data yang diekstrak berasal dari beberapa sumber yang terpisah yang perlu diubah, dikonversikan dan dibuat siap menjadi sebuah format yang sesuai untuk pengkuerian dan analisis.

Tiga fungsi utama yang perlu dijalankan untuk menjadi data siap yakni mengekstrak data, mentransformasi data dan me-*load* data ke dalam media penyimpanan Data Warehouse.Tiga fungsi utama ini yakni *extraction*, *transformation* dan persiapan *loading* data dilakukan dalam *staging area* (wilayah yang berbeda-beda). Data *Staging area*menyediakan sebuah ruang dan area dengan seperangkat fungsi untuk membersihkan, merubah, mengkombinasikan, mengkonversi, menduplikasi dan mempersiapkan *source data* untuk penyimpanan dan penggunaan di dalam Data Warehouse.

a. Data Extraction

Fungsi ini berhubungan dengan sejumlah *Data Sources.* Anda harus menggunakan teknik yang cocok untuk masing-masing data source. *Data Source* bisa jadi berasal dari sumber-sumber mesin yang berbeda-beda dengan format yang beragam. Beberapa bagian data bisa jadi berasal dari DBMS atau dari data hierarkis dan data jaringan atau mungkin beberapa flat file, juga mungkin dimasukkan data dari *spreadsheet* ataupun data set departemen. *Data Extraction* dalam hal ini terbilang cukup kompleks.

*Tool-tool*tersedia di pasaran untuk proses *data extraction.*Tim Implementasi Data Warehouse mengekstraksi *Data Source* ke dalam lingkungan fisik yang terpisah proses menjadikan data ke dalam Data Warehouse menjadi lebih mudah. Dalam sebuah lingkungan yang berbeda, anda perlu mengekstrak *source data* menjadi sekelompok file-file flat, atau *data staging relational database* atau kombinasi dari keduanya.

b. Data Transformation

Dalam setiap implementasi sistem, konversi data adalah sebuah fungsi penting. Sebagai contoh ketika ingin mengimplementasikan sistem operasional seperti

aplikasi majalah, pertama yang harus dilakukan adalah pempopulasian database anda dengan data dari rekod sistem yang terlebih dahulu. Yang boleh jadi dikonversikan dari sistem manual, atau sistem berorientasi file menjadi sistem modern dengan dukungan tabel *relational database.*

Anda juga dapat melakukan sejumlah aktivitas sebagai bagian transformasi data. Yang pertama, membersihkan data yang diekstraksi dari masing-masing sumber. Pembersihan disni dapat berupa koreksi kesalahan penulisan atau bisa jadi memasukkan pemecahan masalah antara kode wilayah dan kode pos dalam *source data* atau boleh jadi juga berhubungan dengan penyediaan nilai *default* untuk elemen-elemen data yang hilang atau menghilangkan duplikat-duplikat ketika membawa didalam data yang sama yang berasal dari *source system* yang berbeda. Data transformasi melibatkan banyak bentuk kombinasi data dari sumber yang berbeda-beda. Anda mengkombinasikan data dari sebuah sumber rekod tunggalatau elemen-elemen data terkait yang berasal dari banyak rekod. Disisi lain transformasi data juga melibatkan pembersihan *source data* yang tidak berguna dan pemisahan rekod sumber luar ke dalam kombinasi baru. Pengurutan dan penggabungan data terjadi dalamskala luas dalam *Data Staging Area.*

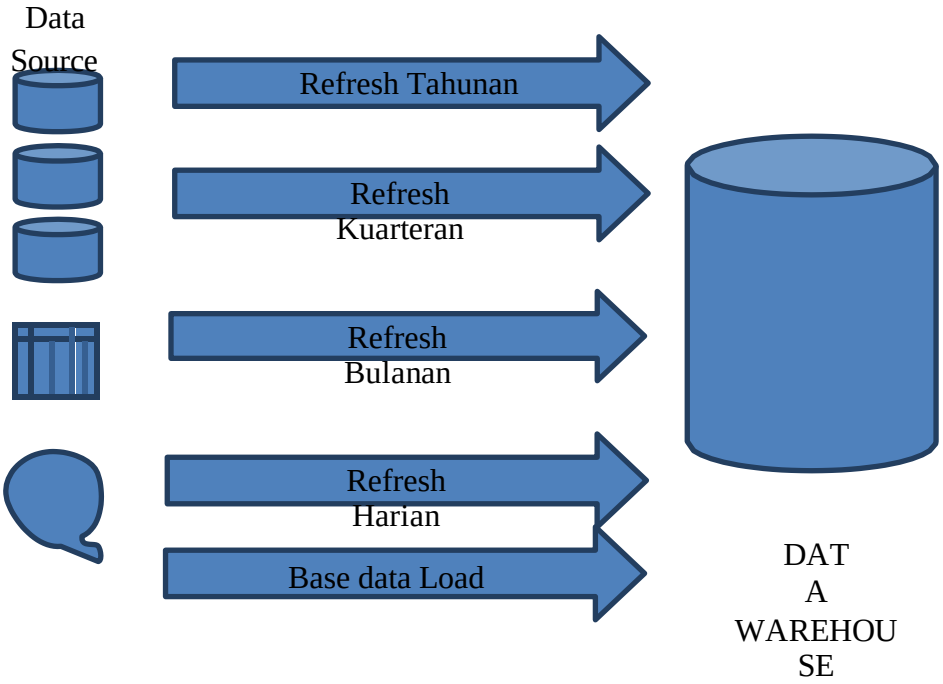
Sistem operasional rantai grosir POS menjaga jumlah unit penjualan dan pendapatan yang dilakukan dalam transaksi tunggal pada konter masing-masing toko. Namun di dalam Data Warehouse tidak dibutuhkan menjaga data pada tingkat yang detil ini. Anda mungkin meringkas total produk pada masing-masing toko untuk hari tertentu dan mempertahankan jumlah total unit penjualan serta pendapatan dalam media penyimpana Data Warehouse. Dalam hal ini maka fungsi *data transformation*termasuk ringkasan yang sesuai.

Setelah fungsi *Data Transformation* berakhir, anda memiliki sekumpulan data teritegrasi yang bersih, standard dan ringkas. Maka Data siap untuk di *load* ke dalam masing-masing Data Set dalam Data Warehouse.

c. Data Loading

Dua kelompok kegiatan membentuk fungsi *Data Loading.* Setelah melengkapi desain dan konstruksi Data Warehouse dan berjalan untuk pertama kalinya, perlu dilakukan *initial loading* Data Warehouse ke dalam media penyimpanan Data

Warehouse. *Load* permulaan memindahkan data bervolume besar menggunakan jumlah waktu yang substansial. Gambar 2.2. mengilustrasikan tipe-tipe umum perpindahan data dari *staging area* menjadi media penyimpanan *Data Warehouse.*



Gambar 2.2. Perpindahan Data ke data Warehouse

2.3. Komponen *Data Storage*

Penyimpanan Data untuk Data Warehouse adalah tempat penyimpanan yang berbeda. Penyimpanan data untuk sistem operasional hanya berisikan data saat ini. Penyimpanan data-data ini berisikan struktur yang benar-benar dalam format nornal untuk pemrosesan yang cepat dan efisien. Sebaliknya, dalam penyimpanan data Data Warehouse, yang harus dilakukan adalah menjaga data dengan volume besar untuk analisis. Lebih jauh lagi data di dalam Data Warehouse harus dijaga dalam struktur yang sesuai untuk analisis dan tidak untuk penerimaan secara cepat seperangkat informasi yang dibutuhkan. Oleh karenanya, media penyimpanan untuk Data Warehouse dipisahkan dari media penyimpanan untuk sistem operasional.

Di dalam database yang mendukung sistem operasional, perbaikan terhadap data terjadi seiring terjadinya transaksi, ketika analis menggunakan data dalam Data Warehouse untuk analisis, yang harus diketahuinya adalah data bersifat stabil dan menggambarkan snapshot pada periode tertentu.

Kebanyakan Data Warehouse juga menggunakan DBMS. Data diekstrak dari media penyimpanan Data Warehouse dijumlahkan dengan banyak macam cara dan ringkasan datanya dijaga dalam Multidimensional databases(MDDB).

2.4. Komponen *Information Delivery*

Siapa pengguna yang membutuhkan informasi dari Data Warehouse? Pengguna awam membutuhkan Data Warehouse tanpa *training*dan oleh karenanya membutuhkan laporan dan kueri, pengguna yang secara kebetulan hanya membutuhkan informasi satu kali tidak secara teratur. Pengguna tipe ini juga membutuhkan informasi yang dipaketkan. Analis bisnis membutuhkan kemampuan untuk melakukan analisis kompleks menggunakan informasi dalam Data Warehouse. *Power user* ingin dapat bernavigasi dengan Data Warehouse memperoleh data yang menarik, membentuk kuerinya, menembus data layer dan membuat kustomisasi laporan serta kueri ad hoc.

Agar dapat menyediakan informasi kepada komunitas pengguna data Warehouse komponen *information delivery* memiliki beberapa metode *information delivery*. Gambar 2.3. menyajikan beberapa metode i*nformation delivery* yang berbeda. *Ad hoc report* adalah laporan yang berarti bagi pengguna awam dan umum. Ketentuan kueri yang komplek, analisis multidimensional (MD) dan analisis statistik untuk memenuhi kebutuhkan analis bisnis dan *power user.* Informasi yang memenuhi kebutuhan *Executive Information System* (EIS) untuk eksekutif senior dan manajer level atas. Beberapa Data Warehouse juga menyediakan data untuk aplikasi *data-mining.* Aplikasi *Data-mining*adalah *knowledge discovery system* dimana algoritma *mining* membantu anda untuk mengetahui tren dan pattern dari penggunaan data anda.

2.5. Komponen Metadata

Metadata dalam sebuah Data Warehouse mirip dengan kamus data atau katalog data dalam sebuah DBMS. Dalam kamus data, informasi seperti struktur data dijaga, informasi mengenai file dan alamatnya, informasi mengenai indeks dan lain sebagainya. Kamus data berisikan data tentang data di dalam database.

Komponen Metadata adalah data mengenai data di dalam Data Warehouse. Metadata di dalam Data Warehouse mirip dengan sebuah sebuah kamus data, namun lebih dari sekedar kamus data.

2.6. Komponen Kontrol dan Manajemen

Komponen ini berada pada posisi yang paling atas dalam arsitektur Data Warehouse. Komponen Kontrol dan manajemen mengkoordinasikan aktivitas dan layanan dalam data Warehouse. Komponen ini mengontrol transformasi data dan data ditransfer kedalam media penyimpanan Data Warehouse. Disisi lain, mencukupi untuk *information delivery* ke pengguna, bekerja dengan DBMS dan memungkinkan data disimpan secara benar dalam tempat penyimpanannya. Juga memonitor pergerakan data ke dalam *staging area* dan dari situ ke dalam media penyimpana Data Warehouse itu sendiri.

3. Metadata dalam Data Warehouse

Metadata dapat diumpamakan *Yellow Pages.* Komponen Metadata bertindak sebagai direktori isi dari Data Warehouse anda. Metadata adalah komponen arsitektur kunci dari Data Warehouse.

Metadata dalam Data Warehouse dibagi menjadi tiga kategori utama, yakni :

- Metadata Operasional, *Source System* ini berisikan struktur data yang berbeda. Elemen-elemen data yang berbeda yang terpilih untuk Data Warehouse memiliki panjang field dan tipe data. Untuk pemilihan data dari *source system* bagi Data Warehouse anda membagi rekod, mengkombinasikan bagian rekod dari beberapa file sumber yang berbeda dan menerapkan skema pengkodean serta panjang field. Metadata operasional berisikan semua informasi mengenai sumber data operasional.
- *Extraction and Transformation Metadata,* berisikan data mengenai ekstraksi data dari sumber data yang dinamakan frekuensi ekstraksi, metode ekstraksi dan aturan bisnis ekstraksi data. Kategori metadata ini berisikan informasi mengenai semua transformasi data yang terjadi dalam *Data Staging area.*
- *End User Metadata,* adalah peta navigasi Data Warehouse. Yang memungkinkan pengguna-akhir untuk menemukan informasi dari Data Warehouse. Metadata pengguna-akhir memungkinkan pengguna-akhir menggunakan terminologi bisnisnya sendiri dan melihat informasi dengan cara dimana mereka secara normal berfikir bisnis.

Metadata sangat penting dalam Data Warehouse, karena :

a. Pertama, ia bertindak sebagai perekat yang menghubungkan semua bagian Data Warehouse.
b. Kemudian, menyediakan informasi mengenai isi dan struktur pengembang.
c. Akhirnya, membuka pintu bagi pengguna-akhir dan menjadikan isinya dapat dikenali dengan terminologinya mereka sendiri

3. Perkembangan Data Warehousing

Data Warehousing telah menjadi *mainstream.* Banyak perusahaan yang memiliki komitmen dalam pembuatan Data Warehouse. Sekitar 90% perusahaan multinasional memiliki Data Warehouse atau juga merencanakan pembuatan Data Warehouse dalam perusahaannya.

Pada hampir semua industri di berbagai bidang, mulai dari ritel, toko hingga lembaga keuangan, dari perusahaan manufaktur hingga departemen pemerintah, mulai perusahaan penerbangan hingga bisnis, Data Warehouse telah merevolusi cara seseorang melakukan analisis bisnisnya dan membuat keputusan strategis. Setiap perusahaan yang memiliki sebuah Data Warehouse, pada dasarnya sedang merealisasikan keuntungan yang sangat besar. Banyak perusahaan-perusahaan ini yang saat ini menggunakan teknologi berbasis web, sedang meningkatkan potensinya dalam menyampaikan informasi vital yang lebih mudah dan lebih besar.

a. Data Warehousing telah menjadi *mainstream*

   Pada wilayah ini, ada empat faktor signifikan yang telah membawa banyak perusahaan menerapkan Data Warehouse:

   i. Kompetisi tinggi
   ii. Peraturan pemerintah
   iii. Kebutuhan untuk merubah proses internal
   iv. Sangat penting untuk kustomisasi pemasaran

   Industri perbankan, telekomunikasi dan ritel adalah industri yang pertama kali menggunakan Data Warehousing, hal ini terjadi pada bisnis telekomunikasi dikarenakan tingginya persaingan, gelombang industri berikutnya yang menerapkan Data Warehousing adalah layanan keuangan, kesehatan, asuransi, manufaktur, farmasi, transportasi dan distribusi.

Di masa kini industri telekomunikasi dan perbankan banyak berinvestasi dalam Data Warehouse. Kurang lebih 15% anggaran teknologi dalam industri-industri dikeluarkan untuk Data Warehouse. Perusahaan dalam industri-industri ini memiliki volume transaksi data yang sangat besar. Data Warehouse mampu mentransformasi volume data besar ini menjadi informasi strategis dalam pengambilan keputusan.

b. Ekspansi Data Warehouse

Walaupun pada awalnya Data Warehouse dikonsentrasikan dalam menjaga ringkasan data untuk analisis level tinggi, namun saat ini dapat dilihat bahwa Data Warehouse yang lebih besar dibangun oleh banyak bisnis yang berbeda-beda. Saat ini perusahaan-perusahaan memiliki kemampuan untuk menangkap, membersihkan, memelihara dan menggunakan sejumlah besar data yang dihasilkan transaksi bisnis mereka.

c. Produk dan solusi vendor

Sebagai seorang profesional di bidang Teknologi Informasi, maka anda akrab dengan vendor database dan produk database. Pada saat yang sama, anda juga akan akrab dengan kebanyakan sistem operasi dan vendornya. Ada ratusan vendor Data Warehousing dan ribuan produk maupun solusi Data Warehousing.

Saat ini perusahaan database tradisional juga memasuki pasar Data Warehousing. Mereka mulai menawarkan solusi Data Warehousing bersama dengan produk database mereka. Di sisi lain, *tool*transformasi dan ekstraksi dipaketkan dengan *Database Management System* (DBMS). Di sisi lain lagi, *tool*pelaporan dan penelaahan dipertajam dengan Data Warehousing. Beberapa vendor database melakukan peningkatan lebih jauh dengan menawarkan produk untuk hal-hal kompleks seperti *tool Data Mining.*

Pasar Data Warehouse secara garis besar dapat dibagi menjadi dua kelompok yakni : kelompok pertama terdiri dari vendor Data Warehouse dan pelayanan produk untuk kebutuhan-kebutuhan Data Warehouse perusahaan dimana semua data enterprise diintegrasikan dan ditansformasikan. Segmen ini dianggap sebagai pasar untuk Data Warehouse strategis. Segmen ini menguasai hampir sepertiga dari total pasar. Segmen kedua lebih longgar dan menebar, terdiri dari data mart departemen, sistem pemasaran database yang terpisah-pisah, dan wilayah sistem pengambilan

keputusan (DSS). Vendor dan produk spesifik mendominasi masing- masing segmen.

4. Trend Yang terjadi

Beberapa pakar berpendapat bahwa teknologi yang telah menggiring Data Warehouse hingga saat ini. Para pakar ini juga berpendapat kita telah melihat kemajuan perangkat lunak yang cukup penting. Data Warehouse telah memicu perubahan besar dalam perkembangan perangkat lunak seperti optimisasi kueri, pengindekan tabel yang sangat besar, peningkatan kompresi data dan*dimensional modeling.*Pertanyaan utama yang harus dipersiapkan jawabannya adalah : Apakah yang harus anda lakukan untuk memperoleh keuntungan dari trend dalam Data Warehouse anda?

a. *Multiple Data Types*

Ketika membangun iterasi pertama dari Data Warehouse, mungkin anda memasukkan data numerik. Namun segera anda akan menyadari bahwa pemasukan data numerik terstruktur tidaklah cukup. Persiapkan untuk mempertimbangkan tipe data yang lain. Secara tradisional, struktur data perusahaan kebanyakan numerik di dalam Data Warehouse mereka. Dari sudut pandang ini maka DSS dapat dibagi menjadi 2 kelompok : Data Warehousing yang berurusan dengan data terstruktur; *knowledge management* yang melibatkan data tidak terstruktur. Contoh : kebanyakan Data *marketing* terdiri data terstruktur yang bernilai numerik. Data *marketing* juga berisikan data tidak terstruktur dalam bentuk gambar. Diumpamakan seorang pengambil keputusan sedang melakukan analisis untuk mencari tipe produk yang paling banyak terjual, setelah menemukan produk tersebut, pengambil keputusan ingin meihat gambar untuk membuat keputusan lebih jauh, bagaimana hal ini dapat diwujudkan? Perusahaan disini menyadari kebutuhan untuk mengintegrasikan baik data terstruktur dan tidak terstruktur di dalam Data Warehouse mereka.

i. Penambahan Data yang tidak terstruktur

Beberapa vendor memasukkan Data yang tidak terstruktur terutama sekali teks dan gambar dengan memeriksa data multimedia sebagaimana tipe data lainnya, yang didefinisikan sebagai bagian relasional data dan disimpan sebagai objek biner (BLOB) hingga ukuran 2GB.

ii. Pencarian Data yang tidak terstruktur

Anda telah menambahkan Data Warehouse dengan menambahkan Data tidak terstruktur. Lalu apalagi yang harus dilakukan? Tentu saja tanpa kemampuan untuk mencari data yang tidak terstruktur, integrasi data akan tidak berarti.

Vendor harus memberikan *search engine* untuk mencari informasi yang dibutuhkan *user* dari data yang tidak terstruktur. Kueri terhadap data gambar adalah contoh mekanisme pencarian gambar.

iii. Data Spasial

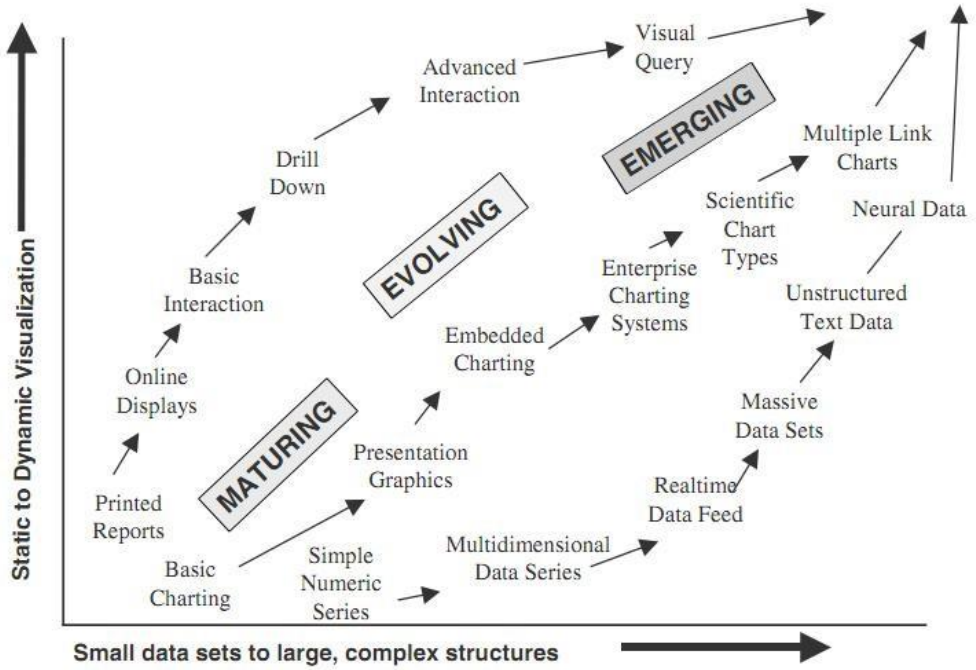
Data Spasial akan meningkatkan nilai Data Warehouse anda. Alamat, blok jalan, kota, kabupaten, kelurahan dan kode pos adalah contoh Data Spasial.

b. Visualisasi Data

Ketika seorang pengguna melakukan kueri dan berharap untuk melihat hasil hanya dalam bentuk daftar hasil atau *spreadsheet*, maka Data Warehouse anda sudah kadaluarsa. Anda harus bisa menampilkan hasil dalam bentuk *chart* dan grafik. Visualisasi data hasil memudahkan proses analisis bagi pengguna ketika pengguna mencari trend dari waktu ke waktu. Visualisasi data membantu pengguna untuk memahami kueri data dengan cepat dan mudah.Tiga trend utama visualisasi data yang telah mengarahkan pembentukan perangkat lunak visualisasi data adalah :

1. Tipe Grafik
2. Visualisasi interaktif, Grafik bersifat interaktif dimana pengguna dapat grafik yang dihasilkan, mengelolanya dan melihat tampilan terbarunya secara online.
3. Visualisasi Result set yang besar dan komplek. Perangkat lunak dengan visualisasi terbaru dapat memvisualisasikan ribuan poin hasil data terstruktur yang kompleks.

Gambar 3.2. menyajikan trend-trend ini. Perhatikan bagaimana teknologi mendewasakan diri, berevolusi dan tumbuh.



Gambar 3.2. Tren Visualisasi Data

ii. Tipe Visualisasi

Kebutuhan *user* saat ini beraneka ragam, *user* bisnis membutuhkan chart pie dan bar, *user* teknis dan saintis membutuhkan *scatter plot* dan *constellation graph.* Analis yang memperhatikan data spasial membutuhkan peta dan representasi tiga-dimensi lainnya. Eksekutif dan manajer yang harus memonitor matrik kinerja, seperti pedoman dijital yang memungkinkan untuk visualisasi matrik seperti *speedometer, termometer* atau lampu lalu lintas.

iii. Advance Visualization Techniques. Kemajuan teknik visualisasi adalah transisi dari *chart* statis menjadi penyajian yang interaktif dan dinamis.

1. *Chart Manipulation, user* dapat memutar *chart* atau secara dinamis dapat mengganti tipe *chart* untuk memperoleh hasil yang lebih diharapkan. Dengan tipe visualisasi yang komplek *user* dapat memilih sebuah titik data dengan *mouse* dan menggerakkan titik tersebut di sekitar *view.*

2. *Drill Down,* pengguna dapat melihat visualisasi data dengan tingkat ketelitian dan lebih detil.

3. *Advance Interaction.* Teknik ini meminimumkan *user interface* yang kurang berguna. Pengguna dibuat langsung dapat men-*double-click* salah satubagian dari visualisasi dan men-drag dan drop representasi entitas data. Atau dibuat lebih sederhana lagi dimana *user* melakukan klik kanan dan menentukan pilihan dari sebuah menu.

c. Parallel *Processing*

i. *Parallel Processing Hardware Options*

ii. *Parallel Processing Software Implementation*

*d. Tool*Kueri

Pada Data Warehouse, seperangkat tool fungsional yang sangat utama adalah seperangkat tool kueri. Kesuksesan Data Warehouse anda bergantung pada *tool kueri.* Banyak vendor Data Warehouse telah meningkatkan kemampuan tool kueri. Fungsi-fungsi yang telah dikembangkan oleh vendor-vendor terkait *tool* kueri antara lain :

i. *Flexible Presentation*

ii. *Aggregate Awareness*

iii. *Crossing Subject Areas*

iv. *Multiple Heterogeneous sources*

v. *Integrasi*

vi. *Mengatasi Keterbatasan SQL*(Overcoming SQL Limitation), menggunakan SQL Extension

e. *Tool Browser*

f. *Data Fusion*

Sebuah Data Warehouse adalah tempat dimana data dari banyak sumber data diintegrasikan untuk menyediakan tampilan yang menyatu pada skala enterprise. Data boleh jadi berasal dari berbagai macam sistem operasi yang berbeda platform atau

mungkin di simpan dalam file flat ataupun DBMS. Dalam penyimpanan Data Warehouse dapat ditemukan juga banyak data yang tidak terstruktur seperti data berbentuk dokumen, gambar, audio dan video.

Berbagai macam data dari banyak sumber yang terpisah ini harus diintegrasikan atau difusikan dan disimpan dalam sebuah Data Warehouse. *Data Fusion* adalah teknologi yang berhubungan dengan merjer data dari sumber data yang berbeda, yang memiliki ruang lingkup luas dan memasukkan merjer data *real-time* dari suatu instrumen maupun sistem monitoring. Banyak penelitian spesifik yang dilakukan terkait teknologi *Data Fusion.* Prinsip dan teknik teknologi *Data Fusion* memiliki sebuah aplikasi langsung ke Data Warehouse.

Data Fusion tidak hanya berhubungan dengan merjer data dari beberapa sumber data, namun juga aplikasi lainnya dalam Data Warehouse. Pada masa kini, dengan teknologi Data Warehouse, kita dapat mengumpulkan data dalam proporsi astronomis,semakin banyak informasi yang disimpan, semakin sulit mencari informasi yang benar pada waktu yang tepat. Teknologi Data Fusion diharapkan yang dapat memecahkan masalah ini. Data Fusion masih menjadi area penelitian,Vendor belum memproduksi tools data Fusion ,yang perlu anda lakukan adalah ikuti perkembangan teknologi.

g. *Multidimensional Analysis*

Saat ini, seluruh lingkungan Data Warehouse menyediakan analisis multidimensional. Ini menjadi bagian integral dari sistem pengiriman informasi data Warehouse. Ketentuan analisis multidimensional bagi *user* secara sederhana berarti bahwa mereka akan dapat menganalisis ukuran bisnis dengan berbagai macam cara. Analisis Multidimensional juga bersinonim dengan Online Analytical Processing (OLAP).

h. *Agent Technology*

Software Agent adalah sebuah program yang mampu menjalankan kegiatan pemrograman atas nama pengguna. Contoh : internet, *Software Agent* dapat digunakan untuk melakukan *sort* atau *filter* email berdasarkan aturan-aturan yang didefinisikan pengguna. Dalam Data Warehouse *Software agent* dapat digunakan untuk memberikan *alert* kepada pengguna mengenai kondisi bisnis yang telah terlebih dahulu

didefinisikan. Beberapa vendor bahkan mengkhusukan pada tool sistem alert. Anda harus mempertimbangkan program *software agent* untuk Data Warehouse anda.

5. Dukungan Infrastruktur terhadap Arsitektur

Ketika berbicara mengenai dukungan infrastruktur terhadap Arsitektur maka pertanyaan pertama yang muncul adalah apa saja elemen-elemen yang dibutuhkan untuk mendukung arsitektur?

Suatu infrastruktur memasukkan banyak elemen-elemen, seperti :

a. Platform basic *computing*, yang memasukkan semua kebutuhan sistem operasi dan perangkat keras (*hardware*)
b. DBMS (*Database Management System*), elemen infrastruktur yang penting.
c. Semua tipe *tool* dan *software*

Infrastruktur Data Warehouse memasukkan semua elemen dasar yang memungkinkan arsitektur untuk diimplementasikan. Sehingga infrastruktur memasukkan beberapa elemen seperti perangkat keras server, sistem operasi, perangkat lunak jaringan, perangkat lunak database, LAN dan WAN, tool vendor untuk setiap komponen arsitektur, orang, prosedur dan pelatihan.

Elemen-elemen infrastruktur Data Warehouse dapat dikelompokkan menjadi dua kategori yakni : infrastruktur operasional dan infrastruktur fisik. Pembedaaan ini penting karena elemen-elemen pada masing-masing kategori memiliki fitur yang berbeda jika dibandingkan dengan kategori lainnya,

a. **Infrastruktur Operasional**

   Salah satu komponen infrastruktur merujuk pada perangkat keras komputer dan perangkat lunak terkait. Anda membutuhkan perangkat keras dan perangkat lunak untuk menjalankan fungsi *data staging* dan memberikan servis yang sesuai. Anda juga membutuhkan tool perangkat lunak untuk menjalankan transformasi data, anda membutuhkan perangkat lunak untuk membuat file output, anda membutuhkan perangkat keras disk untuk menempatkan data dalam file *staging area*. Bagaimana dengan orang-orang yang terlibat menjalankan fungsi-fungsi ini? Apa prosedur dan aturan-aturan bisnis untuk transformasi data? Bagaimana dengan manajemen perangkat lunak untuk memonitor dan mengadministrasikan kegiatan transformasi data?

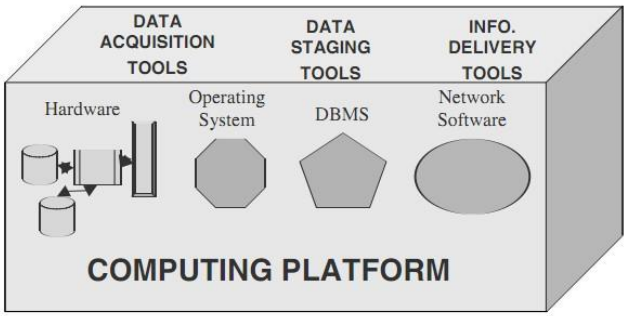
   Infrastruktur Operasional dalam mendukung masing-masing komponen arsitektur terdiri dari :

- Orang
- Prosedur
- Pelatihan
- Perangkat lunak manajemen

Orang dan prosedur disini bukan untuk pengembangan Data Warehouse, namun agar Data Warehouse dapat tetap berjalan, elemen-elemen ini sama pentingnya dengan perangkat lunak dan perangkat keras yang menjaga agar Data Warehouse tetap berjalan. Mereka mendukung manajemen Data Warehouse dan memelihara efisiensinya.

b. Infrastruktur Fisik

Gambar 4.1 menampilkan elemen-elemen utama infrastruktur fisik. Setiap sistem termasuk Data Warehouse harus memiliki keseluruhan platform dimana ia ditempatkan. Secara esensial, platform terdiri dari komponen perangkat keras dasar, sistem operasi disertai dengan software utiliti, jaringan dan software jaringan.



Gambar 4.1. Infrastruktur Fisik

6. Sistem Operasi Dan Perangkat Keras

Sistem operasi dan perangkat keras menciptakan lingkungan komputer untuk Data Warehouse anda.

Seluruh kegiatan ekstraksi data, transformasi, integrasi dan *staging* berjalan pada perangkat keras terpilih dengan sistem operasinya. Ketika anda mentransportasikan data yang telah dikonsolidasikan dan diintegrasikan dari *staging area* ke tempat penyimpana Data Warehouse anda, anda menggunakan perangkat lunak sistem operasi dan perangkat keras server.

Berikut adalah petunjuk pemilihan perangkat keras, yang tidak secara keseluruhan spesifik pada perangkat keras bagi Data Warehouse.

Scalability : Ketka Data Warehouse memiliki makin banyak pengguna, maka dapat dipastikan perangkat keras pilihan anda dapat menangani jumlah kueri dan kompleksitasnya yang semakin meningkat.

Support : Dukungan vendor bersifat krusial bagi pemeliharaan perangkat keras. Pastikan bahwa dukungan dari vendor perangkat keras berada pada peringkat tertinggi yang paling mungkin.

Vendor Reference : Hal yang cukup penting untuk dilakukan yakni pengecekan referensi vendor dengan lainnya menggunakan perangkat keras vendor tersebut. Anda harus mengantisipasi terjadinya Data Warehouse *down* karena *malfungsi* perangkat keras ketika CEO menginginkan beberapa analisis kritis yang harus diselesaikan.

Vendor Stability : Cek stabilitas dan waktu nyala.

Berikutnya kita dapat mempertimbangkanbeberapa kriteria umum untuk pemilihan sistem operasi. Yang pertama, sistem operasi harus kompatibel dengan perangkat keras. Daftar kriterianya adalah sebagai berikut :

Scalability : Scalibility berada pada daftar pertama karena merupakan salah satu fitur utama setiap Data Warehouse. Data Warehouse tumbuh sangat cepat. Begitu juga dengan perangkat lunak dan perangkat keras database, sistem operasi harus dapat mendukung peningkatan jumlah pengguna dan aplikasi.

Security : Ketika banyak *workstation*klien mengakses server, sistem operasi dapat melindungi masing-masing klien dan sumber daya terkait. Sistem operasi harus menyediakan masing-masing klien sebuah lingkungan yang aman.

Reliability : Sistem operasi harus dapat melindungi lingkungan Data Warehouse dari malfungsi aplikasi.

Availability : merupakan akibat langsung dari *reliability*. Lingkungan komputerisasi harus tetap *available* setelah penghentian aplikasi abnormal.

Preemptive Multitasking : Perangkat keras server harus mampu menyeimbangkan alokasi waktu dan sumber data antara banyaknya aktivitas yang berjalan. Sistem operasi juga harus mampu menentukan aktivitas dengan prioritas yang lebih tinggi yang mendahului atau menginterupsi aktivitas lainnya ketika dibutuhkan.

Use multithreaded approach : Sistem operasi harus dapat melayakni banya permintaan yang terjadi secara serentak dengan pendistribusian *thread* ke banyak prosesor dalam

konfigurasi perangkat keras multiprosesor. Fitur ini sangat penting karena konfigurasi multiprosesor adalah arsitektur pilihan dalam sebuah lingkungan Data Warehouse.

Memory Protection : Dalam lingkungan Data Warehouse, ada banyak kueri dalam jumlah besar, yang berarti bahwa banyak kueri yang akan dieksekusi secara serentak. Sebuah fitur proteksi memori dalam sistem operasi mencari mencegah sebuah aktivitas dari merambah ruang memori lainnya.

a. Pilihan Platform

Platform komputer adalah seperangkat komponen perangkat keras, sistem operasi dan jaringan serta perangkat lunak jaringan. Baik merupakan fungsi dari sebuah sistem OLTP atau sistem pengambilan keputusan seperti Data Warehouse, fungsinya harus dijalankan dalam sebuah platform komputer.

i. *Pilihan Single Platform :*adalah pilihan paling sederhana dan mudah. Pada pilihan ini, seluruh fungsi dari ekstraksi data *back-end*hingga pemrosesan kueri front-end dijalankan pada sebuah platform komputer tunggal. Ini adalah pendekatan paling awal yang dikembangkan para *developer* ketika mengimplementasikan Data Warehouse pada mainframe yang sedang eksis, minikomputer atau sebuah server tunggal berbasis UNIX.

Karena seluruh operasi dalam perolehan data, penyimpana data dan area pengiriman informasi terjadi pada platform yang sama, pilihan ini sulit menangani masalah kompatibilitas dan i*nterface*. Tidak dibutuhkan perangkat *middleware,* seluruh tool bekerja dalam sebuah lingkungan komputer tunggal.

ii. *Legacy Platform Stretched to Capacity :*Pada banyak perusahaan, lingkungan komputerisasi yang eksis boleh jadi telah ada sejak beberapa dekade dan sudah memenuhi sesuai kapasitasnya. Linkungan komputernya boleh jadi dimana ia tidak dapat lagi di*upgrade*lebih jauh untuk mengakomodasi Data Warehouse anda.

iii. *Nonavailability of Tools :*Tool perangkat lunak mengambil porsi yang cukup besar dari infrastruktur Data Warehouse. Kebanyakan tool disediakan oleh sejumlah vendor Data Warehouse tidak mendukung linkungan *mainframe* atau minikomputer. Tanpa tool yang cocok dalam infrastruktur maka Data Warehouse akan berantakan.

iv. *Multiple Legacy Platform*

Walaupun kita menduga bahwa warisan komputer *mainframe* atau lingkungan minikomputer dapat diperbesar dengan memasukkan Data Warehouse, fakta praktisnya menunjuk pada sebuah situasi yang berbeda.

v. *Company's Migration Policy*

Anda pasti mengenai banyak keuntungan Arsitektur Client-Server. Anda juga pasti menyadari bahwa setiap perusahaan berubah untuk menerapkan paradigma baru komputerisasi dengan melakukan perubahan model aplikasi dari platform mainframe dan minikomputer. Pada kebanyakan perusahaan, kebijakan penggunaan Teknologi Informasi tidak memungkinkan platform-platform tua untuk abadi. Jika perusahaan anda memiliki kebijakan yang sama, makan anda tidak akan diizinkan untuk menambahkan sistem lain yang signifikasi seperti Data Warehouse pada platform tua.

b. **Hybrid Option**

Setelah menganalisis warisan sistem dan aplikasi yang lebih modern dalam perusahaan anda, maka berikutnya anda dapat memutuskan bahwa pendekatan single-platform tidak dapat bekerja untuk Data Warehouse anda. Ini menjadi alasan banyak perusahaan tidak menjalankannya, disisi lain, jika perusahaan anda harus memilih kategori dimana platform warisan dapat mengakomodasi Data Warehouse maka pendekatan single-platformmenjadi solusi. Untuk lebih jelas, kita akan melihat tahapan *data flow* dan memeriksa pilihan platform.

i. *Data Extraction :* Pada Data Warehouse yang terbaik adalah menjalankan fungsi ekstraksi data dari masing-masing sistem sumber pada platform komputer itu sendiri.

ii. *Initial Reformatting and Merging* : Setelah pembuatan data mentah yang diekstrak dari beberapa sumber, file-file yang diekstrak dari masing-masing sumber di format kembali dan digabungkan menjadi sejumlah file ekstraksi yang lebih kecil. Verifikasi Data ekstrak terhadap laporan *source system* dan rekonsiliasi jumlah rekod input maupun output ada pada tahapan ini. Seperti tahapan ekstraksi, yang terbaik adalah menjalankan tahapan awal penggabungan masing-masing perangkat sumber ekstrak pada platform sumber itu sendiri.

iii. *Preliminary Data Cleansing.* Pada tahapan ini, anda memverifikasi data yang telah diekstrak dari masing-masing *source data* untuk nilai data yanghilang pada field-

field tunggal, memberikan nilai default dan menjalankan edit dasar. Ini adalah tahapan lainnya pada platform komputer dari *source system* itu sendiri. Bagaimanapun, dalam beberapa Data Warehouse tipe pembersihan data ini terjadi setelah data dari semua sumber direkonsiliasikan dan dikonsolidasikan. Pada masing-masing kasus, fitur-fitur dan kondisi-kondisi data dari *source system* anda mendiktekan kapan dan dimana tahapan ini harus dijalankan bagi Data Warehouse anda.

iv. *Transformation and Consolidation :* Tahapan ini berisikan seluruh transformasi data utama dan fungsi integrasi. Anda akan menggunakan tool perangkat lunak transformasi untuk tujuan ini.

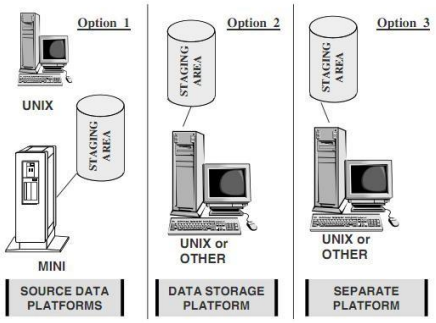
v. *Validation and Final Quality Check :* Tahapan validasi final dan pemeriksaan kualitas adalah kandidat kuat *staging area.* Anda akan menyusun tahapan ini agar terjadipada platform tersebut.

vi. *Creation of Load Images :* Tahapan ini menciptakan *load images* untuk file-file database tunggal dari media penyimpanan Data Warehouse. Tahapan ini hampir selalu terjadi dalam *staging area* dan oleh karenanya pada platform dimana *staging area* ditempatkan.

c. Option for Staging Area

Adalah tempat dimana seluruh data untuk Data Warehouse dikumpulkan dan disiapkan. Platform paling cocok untuk *staging area* tergantung status platform sumbernya, untuk lebih jelasnya mari kita eksplorasi pilihan penempatan *staging area.*

i. **Dalam salah satu Legacy Platform**. Jika kebanyakan warisan Data sources ada pada platform yang sama dan jika kapasitas ekstra tersedia, maka pertimbangkan penjagaan area *data staging* anda dalam *legacy platform.* Untuk pilihan ini, anda akan menghemat waktu dan tenaga dalam memindahkan data berbeda platform ke *staging area.*



Gambar 4.2. Platform *Data Staging Area*

ii. **Pada Platform *Data Storage***. Ini adalah platform dimana DBMS Data Warehouse berada dan berjalan.

iii. Pada Platform yang terpisah. Mungkin saja anda membutuhkan transformasi data kompleks. Hal yang mungkin saja terjadi dimana anda dapat bekerja dengan data anda untuk membersihan dan mempersiapkannya bagi Data Warehouse anda. Dalam hal ini, anda membutuhkan sebuah platform terpisah untuk memilah-milah data sebelum diteruskan ke database.

d. Data Movement Consideration

Pada platform komputer apapun tahapan tunggal *data acquisition* dan *data storage* terjadi, data harus dipindahkan ke platform yang berbeda. Bergantung pada *source* platform perusahaan dan pilihan platform untuk *data staging* dan *data storage,* anda harus menyediakan transportasi data pada platform yang berbeda. Pada masing- maisng pergerakan data lintas platform komputer, tentukan pilihan yang paling cocok untuk lingkungan tersebut. Penjelasan singkat dari pilihan standard tersebut adalah sebagai berikut :

i. Shared Disk

Metode ini kembali ke zaman mainframe. Aplikasi berjalan pada wilayah dan partisi yang berbeda diizinkan untuk berbagi data dengan penempatan data pada suatu disk *share.* Anda dapat mengadaptasi metode ini dari satu tahap ke tahapan lainnya untuk *data acquisition* dalam Data Warehouse anda. Anda harus menentukan *disk storage* dan pengaturannya sehingga masing-masing dari kedua platform mengenali area *disk storage* nya sendiri.

ii. Mass Data Transmission.

Dalam hal ini, tranmisi data berbeda platform terjadi melalui *port data*. Port data adalah perangkat *interplatform* yang sederhana yang memungkinkan kuantitas data massal ditransportasikan dari satu platform ke yang lainnya. Masing-masing platform harus dikonfigurasikan untuk menangani transfer melalui port. Pilihan ini membutuhkan komponen perangkat keras, perangkat lunak dan jaringan khusus. Juga dibutuhkan bandwith jaringan yang cukup dalam pemrosesan data skala besar.

iii. Real-Time Connection

Dalam pilihan ini, dua platform menciptakan koneksi secara real time sehingga sebuah program dapat berjalan pada sebuah platform yang mampu menggunakan sumber daya platform lainnya. Sebuah program pada sebuah platform dapat melakukan penulisan terhadap *disk storage* lainnya. Kegiatan yang berjalan pada sebuah platform dapat melakukan penjadwalan aktivitas dan event lainnya.

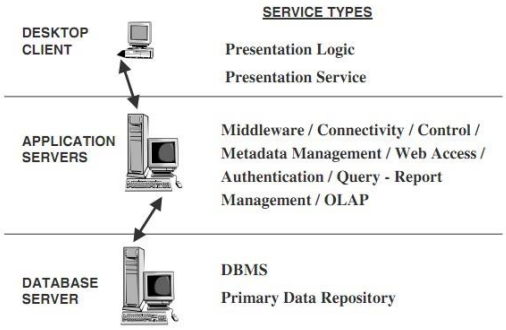
iv. Manual Methods

Pilihan ini sederhana dan mudah. Sebuah program pada sebuah platform melakukan penulisan terhadap sebuah media eksternal seperti tape dan disk. Program lainnya pada platform penerima membaca data dari media eksternal

e. Client-Server Architecture for the Data Warehouse

Walaupun platform minikomputer dan mainframe lebih dahulu diimplementasikan untuk Data Warehouse, namun saat ini, ketika semakin besar, Data Warehouse dikembangkan dengan menggunakan arsitektur client-server. Kebanyakan dari ini

adalah *multitiered,* arsitektur client/server generasi kedua. Gambar 4.4. mengilustrasikan arsitektur client/server untuk implementasi Data Warehouse.



Gambar 4.4. Arsitektur client/server Data Warehouse

f. Considerations for Client Workstations

Ketika anda siap untuk mempertimbangkan konfigurasi mesin workstation, anda akan lebih cepat menyadari bahwa anda butuh untuk memisahkan tipe pengguna. Kita hanya mempertimbangkan kebutuhan workstation terkait pengiriman informasi dari Data Warehouse. Pengguna diharapkan puas dengan sebuah mesin yang dapat berjalan pada sebuah browser web untuk mengakses laporan HTML. Analisis serius, disisi lain, membutuhkan mesin workstation yang lebih besar dan powerful. Tipe pengguna lainya antara dua keekstriman ini membutuhkan sebuah model layanan. Gunakan ceklis dibawah ini ketika mempertimbangkan workstation :

- Sistem operasi Workstation
- Processing power
- Memori
- Disk Storage
- Transportasi Data dan jaringan
- Dukungan tool

g. Options as The Data Warehouse Matures

Anda mungkin dapat menyimpulkan bahwa pilihan platform telah benar dan pilihan awal ditentukan. Hal yang menarik untuk dicatat bahwa Data Warehouse pada masing-masing enterprise yang telah mapan, maka susunan platformnya juga berevolusi. Data Staging dan Data Storage dapat berada pada platform komputer yang sama.

7. Dimensional Modeling

Istilah atau nama *Dimensional Modeling* berasal dari dimensi bisnis yang kita perlu tuangkan ke dalam model data logical. Pemodelan ini adalah teknik desain untuk menstrukturkan dimensi bisnis dan matrik yang dianalisis bersama dengan dimensi-dimensi . Model juga memberikan kinerja tinggi untuk analisis dan kueri.

Bisnis bagi manajer adalah terminologi dimensi bisnis. Contoh seorang pimpinan *marketing* tertarik mengetahui jumlah pendapat bulanan, di dalam sebuah divisi, berdasarkan demografi pelanggan yang dilakukan sales untuk jenis produk tertentu di masa yang sudah terjadi. Dalam hal ini dimensi bisnis adalah bulan, divisi, demografi, sales, dan jenis produk. Pendapatan adalah *fact* yang ingin diketahui pimpinan *marketing.*

Dimensional Modeling adalah teknik pemodelan favorit dalam Data Warehouse. Di dalam *Dimensional Modeling,* sebuah model tabel dan relasi digunakan untuk tujuan pengoptimalan kinerja kueri pengambilan keputusan dalam database relasional, relatif terhadap pengukuran atau seperangkat pengukuran proses bisnis yang akan dimodel. Sebaliknya model E-R konvensional digunakan untuk (1) menghilangkan pengulangan (*redundan*) dalam model data, (2) memfasilitasi perolehan rekod individual yang memiliki sejumlah identifier tertentu dan (3) mengoptimalkan kinerja OLTP.

Pada sebuah toko ritel, pengukuran yang penting atau *fact* adalah unit penjualan. Dimensi bisnis bisa jadi waktu, promosi, produk atau toko. Untuk suatu perusahaan asuransi, pengukurannya atau *fact* adalah klaim dan dimensi bisnisnya agen, kebijakan, pihak yang diasuransikan,status dan waktu.

Praktisi *Dimensional Modeling* menggunakan pendekatan model data logik dengan pemilihan proses bisnis yang akan dimodel dan kemudian memutuskan masing-masing rekod dalam “fact table”. *Fact table* adalah fokus analisis dimensional, adalah tabel dengan kueri berdimensi untuk memberikan seperangkat solusi. Kriteria untuk segmentasi ada di dalam satu atau lebih “tabel dimensi” dimana *primary key* tunggalnya menjadi *foreign keyfact table* yang berelasi dalam desain *dimensional model.*

Contoh dibawah ini menunjukkan bahwa dimensi bisnis dan relevansinya terhadap industri serta subjek untuk analisa. Begitu juga dimensi waktu bersifat penting untuk semua industri – yang hampir semua analisis bisnis jalankan di setiap waktu.

8. TABEL DIMENSIONAL

Ketika suatu dimensi bisnis diabstraksikan dan direpresentasikan dalam sebuah tabel database, ia dikenal dengan tabel dimensional. Sebuah dimensi dapat dipandang sebuah sebuah entitas. Sebuah tabel dimensional menyediakan deskripsi tekstual dari sebuah dimensi bisnis melalui atribut-atributnya.

Tabel dimensional cenderung secara relatif dangkal dalam hal jumlah baris-barisnya, namun dilengkapi banyak kolom-kolom. Sebuah tabel dimensional memiliki sebuah primary key tunggal dan telah didenormalisasi.

Atribut tabel Dimensional memainkan sebuah peran vital dalam pemrosesan kueri dan pelabelan laporan. Kekuatan sebuah Data Warehouse secara langsung proporsional berhubungan dengan kualitas dan kedalaman atribut-atribut dimensinya.

9. FACTS

*Fact*adalah pengukuran yang diperoleh dari kejadian yang ada dipasaran, merupakan sumber pengetahuan dan observasi.Seorang pelanggan membeli sebuah produk pada suatu lokasi tertentu pada waktu tertentu. Persilangan dari keempat dimensi yang terjadi ini menciptakan sebuah penjualan. Penjualan dapat dideskripsikan sejumlah uang yang diterima, jumlah item terjual, berat produk yang dikirimkan dan lain-lain. Fact digunakan untuk merepresentasikankan sebuah ukuran bisnis.

Dalam sebuah Data Warehouse, *fact* didefinisikan sebuah persilangan dari dimensi-dimensi yang menggunakan entitas-entitas dasar transaksi bisnis. Untuk menunjukkan persilangan lebih dari tiga dimensi dalam sebuah diagram bukan perkara mudah, namun *fact* dalam sebuah Data Warehouse boleh jadi berasal dari banyak dimensi.

10. TABEL FACT

Sebuah tabel adalah tabel primer di dalam model dimensional dimana pengukuran kinerja secara numerik dari bisnis disimpan. Ada banyak pengukuran kinerja atau *fact* dalam sebuah tabel *fact.* Sebuah baris dalam sebuah tabel *fact* berhubungan dengan pengukuran. *Fact* yang paling berguna dalam sebuah tabel *fact* adalah numerik dan tambahan.

Semua tabel-tabel *fact* memiliki dua atau lebih *foreign key* yang berhubungan dengan *primary key* tabel dimensi. Ketika semua key di dalam tabel *fact* cocok dan berpasangan

dengan *primary key*-nya secara benar dengan tabel dimensi maka dapat dikatakan tabel tersebut memenuhi *referential integrity*. Tabel *fact* diakses melalui tabel dimensi yang diikut sertakannya.

Tabel *fact* sendiri secara umum memiliki *primary key* yang disusun seperangkat *foreign key*. *Key* ini dikenal dengan sebuah key *composite* atau *concatenated*(berentetan) . Setiap tabel *fact*dalam sebuah model dimensional memiliki sebuah key composite, dan sebaliknya setiap tabel yang memiliki sebuah key *composite* adalah sebuah tabel *fact.*

Cara lain untuk mengungkap hal ini adalah di dalam sebuah *dimensional model,* setiap tabel yang mengekspresikan relasi banyak-ke-banyak sudah pasti sebuah tabel *fact.* Semua tabel lainnya adalah tabel-tabel dimensi.

Tabel Fact Penjualan Harian
Key Tanggal(FK)
Key Produk(FK)
Key Toko(FK)
Jumlah terjual
Total Harga

11. Star Schema

Suatu model yang menggabungkan dimensi dan *fact* bersamaan dikenal dengan dimensional model. Di dalam model ini, tabel *fact* terdiri dari pengukuran numerik yang disertakan ke seperangkat tabel dimensi yang dipenuhi atribut deksriptif.

Di dalam model ini, Tabel *fact* berada di tengah dan tabel dimensi bergelantungan disekitarnya seperti sebuah bintang. Struktur dengan karakteristik demikian seringkali dikenal dengan *star schema.* Saat id pelanggan, id produk dan periode waktu digunakan untuk menentukan baris-baris manakah yang dipilih dari tabel *fact,* cara pengumpulan data demikian dikenal dengan istilah *star schema join.*

Dimensional model sederhana dan simetris, datanya lebih mudah difahami dan dinavigasikan. Setiap dimensi ekuivalen; semua dimensi memiliki entri poin yang sama secara simetris ke dalam tabel *fact.* Kesederhanaan juga menguntungkan kinerja, sedikit join penting bagi pemrosesan kueri. Sebuah mesin database mampu menciptakan asumsi yang kuat mengenai tabel dimensi dengan indeksnya.

Dengan dimensional model juga dapat ditambahkan secara lengkap dimenasi baru ke skema bersamaan dengan nilai tunggal dimensi tersebut didefinisikan untuk masing-masing baris *fact* yang ada.

Cara lain memahami memahami kesederhaaan *star schema* adalah dengan melihat bagaimana dimensi dan *fact* berkontribusi terhadap laporan. Atribut tabel dimensi menawarkan pelabelan laporan, sebagaimana tabel *fact* menawarkan nilai numerik laporan. *data cube*. Hal ini memungkinkan data dimodel dan dilihat dalam banyak dimensi, konsep ini juga dikembangkan berdasarkan *fact* dan dimensi. *Data Cube* dapat didefinisikan sebagai persilangan dimensi-dimensi yang menyediakan beberapa *fact* yang menarik bagi bisnis. Cube cocok bagi pemrosesan OLAP(pemotongan dan pembagian sebuah dimensi bisnis),dapat dikomparasikan dengan *star schema* yang cocok untuk pemrosesan kueri. *Data Cube*dapat diterjemahkan menjadi *star schema.* Bagaimanapun, agregasi data level tinggi secara efisien disimpan sebagai cube; yang telah dikalkulasikan. Cube klasik adalah pelanggan berdasarkan produk berdasarkan waktu (atau tempaty), dan merupakan sebuah cube 3-dimensi (3-D).

Walaupun dalam pandangan kita kubus memiliki struktur geometris 3-D, dalam Data Warehouse data cube dapat berupa n-dimensional. Untuk memperoleh pemahaman yang lebih baik mengenai data cube, kita dapat memulai dengan memahami contoh sebuah data cube 2-D seperti tabel atau spreadsheet untuk data penjualan per kuarter(dimensi waktu) pada item-item tertentu (dimensi produk) untuk sebuah lokasi tertentu. Ukuran Fact adalah nilai uang dalam ribuah dolar yang terjual., perhatikan gambar dibawah ini untuk lebih memahaminya

location = "Vancouver"				
	item (type)			
time (quarter)	home entertainment	computer	phone	security
Q1	605	825	14	400
Q2	680	952	31	512
Q3	812	1023	30	501
Q4	927	1038	38	580

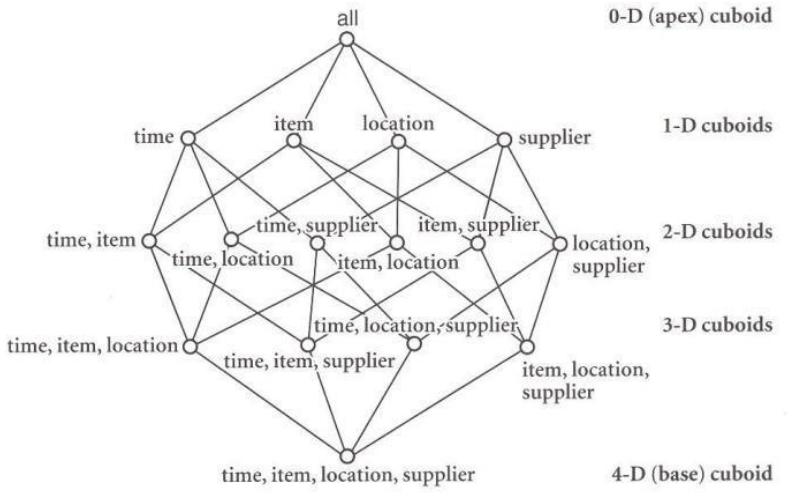
Gambar diatas adlah bentuk 2-D data penjualan untuk seluruh elektronik berdasarkan dimensi waktu dan item, dimana penjualan berasal dari cabang yang berada di kota “vancouver”.

Untuk melihat data penjualan dalam bentuk tiga dimensi(lokasi), kita dapat menambahkan data penjualan 2-D untuk lokasi lainnya. Secara konseptual kita dapat melihat data ini dalam bentuk data cube 3-D seperti gambar dibawah ini :

Gambar diatas adalah representasi *data cube* dari data dalam tabel, menurut dimensi waktu, item dan lokasi. Andaikan kita ingin melihat data penjualan dengan bentuk empat dimensi, seperti *suplier*. Tampilan 4-D untuk dilihat pasti rumit, namun bagaimanapun, kita dapat membayangkan kubus 4-D sebagai rangkaian kubus 3-D, sebagai digambarkan dibawah ini. Jika kita menggunakan cara dibawah ini, maka kita dapat menampilkan data sebanyak n-D dalam sebuah rangkaian kubus (n-1) D.

Data Cuba adalah sebuah konsep untuk penyimpana data multidimensional, penyimpnana fisik actual dari data demikian boleh jadi berbeda dengan representasi logiknya.

Dalam literatur Data Warehouse, kubus 1-D, 2-D,3-D dst dikenal dengan sebutan *cuboid*. Dengan seperangkat dimensi-dimensi, kita dapat membuat seperangkat cuboid, masing-masing menunjukkan data pada level ringkasan data yang berbeda. Cuboid yang menangani level ringkasan terenda dikenal dengan *base cuboid* . sebagai contoh, cuboid 4-D dibawah ini adalah cuboid dasar untuk dimensi waktu, item, lokasi dan suplier tertentu. Puncak cuboid ditunjukkan oleh semua.



14. Hierarki dalam Dimensi

Dalam sebuah Data Warehouse atau Data Mart, Pengukuran disimpan dalam tabel *fact* dengan tingkat ketelitian sedemikian rupa sehingga pengguna dapat melacak ringkasan berdasarkan levelnya. Hal ini dikenal dengan agregasi. Sebagai contoh, jika data penjualan dalam sebuah toko grosirdijaga dalam level sebuah pelanggan tunggal yang membeli item tertentu pada hari tertentu dalam toko tertentu, sehingga kita dapat meringkas atau menjumlahkan data untuk hari, minggu,

bulan, kuartal dan tahun tertentu; dan semua hal ini adalah untuk sebuah toko, zona, wilayah dan negara sebagaimana beradasarkan produk, kelompok produk, departemen dan seterusnya.

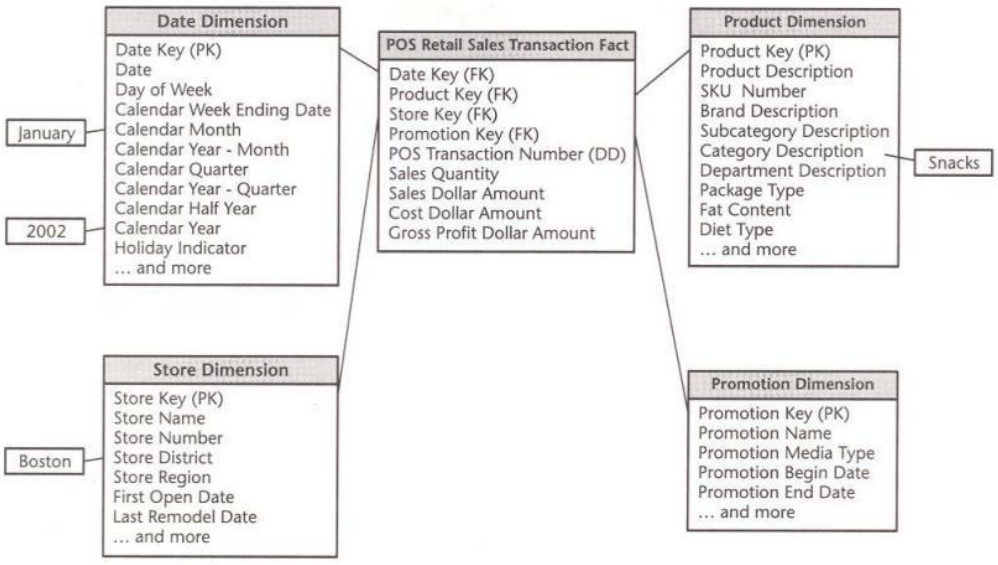
Hanya data penjualan pada level terendah yang dijaga dalam tabel *fact*, namun deskripsi dari berbagai level data di jaga dalam tabel dimensi,sehingga tool yang sesuai dapat digunakan untuk meringkas data dalam level yang bervariasi.

Sebuah hierarki mendefinisikan sebuah urutan pemetaan dari seperangkat konsep low-level hingga level yang lebih tinggi, konsep level yang lebih umum. Bayangkan sebuah hierarki dimensi Lokasi. Jika Kota adalah level terendah dari hierarki , maka semua kota dapat dipetakan ke level yang lebih tinggi lagi, yakni provinsi, dan semua provinsi dapat dipetakan ke level yang lebih tinggi lagi yakni negara. Dst.

Level dimensional membentuk struktur seperti sebuah pohon dan anggota level terendah dari hieararki dikenal dengan *leaf*. Hanya ada satu anggota pada level paling atas. Sebuah dimensi tidak dapat eksis tanpa anggota *leaf*nya, namun hal yang mungkin untuk memiliki sebuah dimensi tanpa anggota leaf – yakni yang hanya memiliki sebuah level.

15. **Pengimplementasian Hierarki Dimensional**

Hierarki dimensional disimpan sebagai atribut dalam tabel dimensi, dan seluruh hierarki terkait disimpan dalam sebuah tabel dimensi tunggal. Deskripsi untuk masing-masing level hierarki dijaga dalam metadata multidimensional. Sebagai contoh, tanggal, hari, bulan, dan tahun disimpan dalam sebuah dimensi tanggal; sementara produk, merek, kategori dan departemen disimpan dalam dimensi produk. Contoh dibawah ini mengilustrasikan skema database toko ritel dan tanggal yang terkait maupun tabel dimensi produk,



16. Penggunaan Hierarki Dimensional

Hierarki dalam dimensi digunakan untuk pemilihan dan agregasi data dengan level yang ditentukan sesuai kebutuhan. Tabel *fact* berisikan data hanya level paling rendah dalam hierarki.

Data pada level yang lebih tinggi diperoleh melalui agregasi data *fact* level paling rendah untuk instan yang sama dari sebuah atribut level dimensional. Pada contoh yang telah diberikan, jika kita ingin menemukan total Jumlah Penjualan serta Total Harga Penjualan untuk masing dari dua departemen, *Bakery* dan *Frozen Food,* Kita pertama kali memilih Bakery dan Frozen Food dari tabel Dimensi Produk dan kemudian menambahkan seluruh nilai kuantitas penjualan serta Total Harga dari tabel *fact* yang berhubungan dengan kedua produk tersebut. Hal ini membutuhkan penambahan secara terpisah, nilai *fact* untuk key produk = 1,2,3 dan 4 serta key produk = 5,6,7,8 dan 9 bagi seluruh nilai yang mungkin dari key lainnya dalam tabel *fact.*

Deskripsi Departemen	Kuantitas Penjualan	Total Harga Penjualan
Bakery	5,088	$12,331
Frozen Food	15,565	$31,776

Selain agregasi berdasarkan deskripsi produk, jika kita ingin melihat lebih rinci dekripsi merk dari produk, kita rancang Deskripsi Produk dan Deskripsi Merk dari Dimensi Produk dan kemudian memilih seluruh kuantitas penjualan serta jumlah Total Harga Penjualan dari tabel *fact* dan menambahkannya.

Department Description	Brand Description	Sales Dollar Amount	Sales Quantity
Bakery	Baked Well	$3,009	1,138
Bakery	Fluffy	$3,024	1,476
Bakery	Light	$6,298	2,474
Frozen Foods	Coldpack	$5,321	2,640
Frozen Foods	Freshlike	$10,476	5,234
Frozen Foods	Frigid	$7,328	3,092
Frozen Foods	Icy	$2,184	1,437
Frozen Foods	QuickFreeze	$6,467	3,162

17. Operasi OLAP : Kueri Data Multidimensional

Dalam model multidimensional, data diorganisasikan menjadi dimensi banyak, dan masing-masing dimensi berisikan level abstraksi yang banyak yang didefinisikan oleh hierarki-hierarki. Organisasi ini menyediakan bagi pengguna kemampuan untuk melihat data dari perspektif tyang berbeda. Sejumlah operasi *data cube* eksis untuk terwujudnya sudut pandang yng berbeda, memungkinkan pnegkuerian dan analisis data yang interaktif. Berikut adalan beberapa operasi OLAP untuk Data Multidimensional.

Dalam penjelasan ini akan digunakan contoh sebuah kubus (cube) yang berisikan dimensi lokasi, waktu dan item dimana lokasi diagregasikan dengan respek terhadap data kota, waktu

diagregasikan dengan respek terhadap kuartal dan item diagregasikan dengan respek terhadap tipe.

Roll-Up : Operasi roll-up atau drill-up menjalankan agrehasi pada sebuah *data cube,* baik dengan menaiki sebuah hierarki data untuk sebuah dimensi atau dengan pengurangan dimensi. *Roll-up* dengan pengurangan dimensi berarti bahwa agregasi dijalankan hingga level atas dimensi. Sebagai contoh, jika hierarki lokasi berisikan tiga level, kota provinsi negara, maka reduksi dari dimensi lokasi berarti hasil data *fact* diringkaskan terhadap kotanya dan kemudian provinsinya.

Drill-Down: kebalikan roll-up. Drill down menavigasikan dari data rinci yang sedikit hingga yang lebih detil, hal ini dapat dilakukan dengan menuruni hierarki untuk sebuah dimensi atau memperkenalkan dimensi tambahan. Penambahan sebuah dimensi baru berarti tabel *fact* pasti berisikan (atau ditambahkan) data di dalam dimensi tersebut.

Slide and Dice : Operasi *slice* menjalankan seperangkat seleksi pada sebuah dimensi dari *cube* tertentu yang menghasilkan sebuah *subcube.* Sebagai contoh, kita dapat memilih seluruh data penjualan untuk seluruh kota dan item pada kuarter tertentu = Q1

Operasi dice mendefinisikan sebuah *subcube* dengan menjalankan sebuah seleksi pada dua atau lebih dimensi. Sebagai contoh, kita dapat melakukan slice pertama kali pada *waktu* untuk memasukkan penjualan pada beberapa kuartal. Dan kemudian pada *location* untuk memasukkan penjualan pada beberapa kota.

Pivot(Rotate) : Pivot adalah sebuah operasi visualisasiyang merotasikan data axis untuk memberikan sebuah alternatif presentasi data

18. **Pengantar mengenai *Extraction, Transformation* dan *Loading***

Data Warehouse dibutuhkan selalu untuk melayani tujuan dan kepentingan analis bisnis. Untuk melakukan hal ini maka data yang berasal dari satu atau beberapa sistem operasi harus diekstrak dan dikopi ke dalam Data Warehouse. Tantangan dalam lingkungan Data Warehouse adalah untuk mengintegrasikan, menyusun kembali dan mengkonsolidasikan volume data yang cukup besar melalui banyak sistem, yang oleh karenanya memberikan satu kesatuan informasi baru yang menjadi basis bagi *business intelligence.*

Proses Ekstraksi data dari beberapa sumber data dan memasukkannya ke dalam Data Warehouse dikenal dengan sebutan ETL, yang merupakan singkatan dari Extraction, Transformation dan Loading. Dimana ETL menunjukkan sebuah proses luas yang terdiri dari tiga tahapan. Akronim ETL nampak terlihat sederhana karena menghilangkan fase transportasi dan masing-masing fase

proses lainnya yang berbeda-beda. Walaupun, keseluruhan proses tersebut dikenal dengan sebutan ETL.

Metodologi dan kegiatan ETL dikenal dengan baik selama beberapa tahun, dan tidak hanya bersifat unik digunakan dalam lingkungan Data Warehouse: sistem database dan aplikasi berhak cipta yang menjadi *backbone* proses IT skala Enterprise. Data telah di*share* antara aplikasi atau sistem, mencoba untuk mengintegrasikannya, memberikan paling tidak dua aplikasi gambaran yang sama mengenai dunia. Data *Sharing* demikian paling sering dikenal dengan mekanisme yang disebut ETL.

19. **Ekstraksi dalam Data Warehouse**

Ekstraksi (*Extraction*) adalah operas ekstraksi data dari sebuah sistem sumber untuk digunakan lebih jauh dalam lingkungan Data Warehouse. Tahapan ini adalah yang paling pertama dalam proses ETL. Setelah Ekstraksi, data ini akan ditransformasikan dan di-*load* ke dalam Data Warehouse.

Pendesainan dan Pembuatan proses Ekstraksi adalah satu kegiatan yang paling sering menyita waktu di dalam proses ETL dan dalam keseluruhan proses Data Warehouse. Sistem sumber sangat proses komplek serta didokumentasikan secara buruk, sehingga menentukan data yang dibutuhkan untuk diekstrak menjadi sulit. Data diekstrak tidak hanya sekali namun beberapa kali dalam suatu periode untuk mensuplai data ke dalam Data Warehouse dan menjaga agar up-to- date. Lebih jauh lagi, Sistem sumber tidak dapat dimodifikasi atau bahkan kinerja dan ketersediaannya tidak dapat diatur untuk mengakomodasi kebutuhan proses ekstraksi Data Warehouse.

Metode Ekstraksi dalam Data Warehouse adalah beberapa bentuk dan benar-benar tergantung pada Sistem sumber juga kebutuhan bisnis dalam lingkungan target Data Warehouse. Tidak ada kemungkinan untuk menambahkan logika tambahan ke dalam sistem sumber dalam proses ekstraksi data yang bersifat inkremental sehubungan dengan kinerja atau beban kerja sistem yang meningkat.

Ada dua bentuk Metode Ekstraksi logical.

i. Ekstraksi Full(*Full Extraction*)

   Data diekstrak secara lengkap dari sistem sumber. Ekstraksi ini melibatkan seluruh data yang sedang tersedia dalam sistem sumber. Data sumber disediakan dan tidak dibutuhkan logika informasi tambahan (seperti *timestamp*) yang dibutuhkan pada situs sumber. Sebuah contoh ekstraksi penuh adalah ekspor file dari sebuah tabel yang berbeda atau kueri remote SQL yang membaca sumber data lengkap

ii. Ekstraksi Inkremental (*Incremental Extraction*)

Pada poin waktu tertentu, hanya data yang memiliki histori data akan diekstrak. *Event* ini adalah proses ekstraksi yang dilakukan paling akhir atau sebagai contoh sebuah *event* bisnis yang komplek seperti hari *booking* terakhir dari suatu periode fiskal. Informasi ini juga dapat disediakan oleh data sumber itu sendiri seperti sebuah kolom aplikasi, merefleksikan *time-stamp* yang paling akhir berubah atau sebuah tabel yang berubah dimana sebuah mekanisme tambahan yang sesuai menjaga track perubahan selain transaksi yang permulaan. Dalam banyak hal, menggunakan metode ini berarti menambah logika ekstraksi ke dalam sistem sumber.

Kebanyakan Data Warehouse tidak menggunakan teknik menangkap perubahan sebagai bagian proses ekstraksi. Sebalaiknya seluruh tabel dari sistem sumber diekstrak ke Data Warehouse atau *staging area*, dan tabel-tabel ini dibandingkan dengan yang telah diekstrak sebelumnya dari sistem sumber untuk mengidentifikasi data yang berubah. Pendekatan ini boleh jadi tidak memiliki pengaruh yang signifikan terhadap sistem sumber, namun secara jelas dapat menciptakan sebuah beban yang dapat dipertimbangkan pada proses Data Warehouse, terutama sekali jika volume data sangat besar.

Bergantung pada metode ekstrasi logikal yang dipilih dan kapabilitasnya serta restriksi pada sisisumber data, dta yang diekstrak secara fisik diekstrak dengan dua mekanisme. Data dapat diekstrak online dari sistem sumber atau dari sebuah struktur offline. Struktur offline demikian boleh jadi ada atau dapat dihasilkan oleh sebuah fungsi ekstraksi.

Ada dua metode ekstraksi fisik (*physical extraction*)

iii. *Online Extraction*

Data diekstrak langsung dari sistem sumber itu sendiri. Proses ekstraksi dapat berhubungan secara langsung dengan sistem sumber untuk mengakses tabel sumber atau ke sebuah sistem perantara yang menyimpan data dengan sebuah cara yang dikonfigurasikan terlebih dahulu (sebagai contoh log atau tabel perubahan). Dengan catatan bahwa sistem perantara secara fisik tidak berbeda dari sistem sumber.

iv. *Offline Extraction*

Data tidak diekstrak secara langsung dari sistem sumber namun dibatasi secara eksplisit diluar sistem sumber orisinil. Data telah memiliki struktur atau telah dibuat melalui prosedur ekstraksi.

Beberapa struktur yang harus dipertimbangkan antara lain :

- *Flat file*
- *Dump File,* informasi mengenai objek yang dimasukkan atau tidak dimasukkan, bergantung pada *utility* yang dipilih.
- *Log Archive* dan *Redo*
- *Transportable Tablespaces,* cara ekstrak dan memindahkan data bervolume besar antar Database.

Beberapa contoh Ekstraksi Data Warehouse, antara lain :

v. Ekstraksi menggunakan Data File

Kebanyakan Sistem database menyediakan mekanisme pengeksporan atau pelimpahan data dari format database internal menjadi file flat. Ekstraksi dari sistem mainframe seringkali menggunakan program COBOL, namun kebanyakan database, sebagaimana vendor perangkat lunak pihak ketiga, menyediakan ekspor atau pelimpahan utility.

Ekstraksi Data tidak berarti bahwa seluruh struktur database dilimpahkan dalam *flat file*. Dalam banyak hal, boleh jadi hal ini sesuai atau cocok untuk melimpahkan seluruh tabel database atau objek. Dalam hal lain, bisa jadi yang cocok hanya melimpahkan seperangkat

tabel yang diberikan seiring perubahan pada sistem sumber. Jika dalam sistem Database oracle atau SQL Server, ada beberapa alternatif yang tersedia untuk mengekstrak Data menjadi File antara lain :

a. Ekstraksi menjadi *Flat File* menggunakan SQL*Plus

   Adalah teknik dasar dalam ekstraksi data yang paling banyak digunakan dan langsung menciptakan hasil kueri ke dalam bentuk file. Sebaai contoh mengekstraksi flat file country_city.log dengan menggunakan delimiter *piping* '|' antar koom data,yang berisikan daftar kota di US dalam tabel *countries* dan *customers,* maka kueri yang dapat dijalankan adalah sbb:

```
SET echo off SET pagesize 0 SPOOL country_city.log
SELECT distinct t1.country_name ||'|'|| t2.cust_city
FROM countries t1, customers t2 WHERE t1.country_id = t2.country_id
AND t1.country_name= 'United States of America';
SPOOL off
```

b. Ekstraksi ke File Ekspor menggunakan Utility Export

   Utility Export memungkinkan Tabel (termasuk data) untuk diekspor menjadi file Ekspor Database. Yang menerangkan ekstraksi dari hasil kueri SQL. Ekspor berbeda dengan metode sebelumnya dengan perbedaan sbb:

   - File Export berisikan metadata seperti data. File Export berisikan tidak hanya Data mentah dari sebuah tabel, melainkan juga informasi bagaimana membuat tabel kembali, secara potensial termasuk indeks, constraint, grant dan atribut lainnya yang berhubungan dengan tabel.
   - Sebuah File export tunggal boleh jadi berisikan seperangkat objek tunggal, banyak objek database atau bahkan keseluruhan skema.
   - Export tidak dapat secara langsung digunakan untuk mengeksport hasil kueri komplek. Export dapat digunakan hanya untuk mengekstrak seperangkat objek database yang berbeda.
   - Hasil dari utility export dapat diproses menggunakan utility

import. c. Ekstraksi ke File Eksport menggunakan Tabel Eksternal

Selain menggunakan utility Export, eksternal tabel juga dapat digunakan untuk mengekstrak hasil menggunakan operasi SELECT. Data disimpan dalam platform bebas. Contoh ekstraksi hasil operasi join yang diparalelkan menjadi empat file. Jika menggunakan Oracle, maka format yang digunakan ORACLE_DATAPUMP

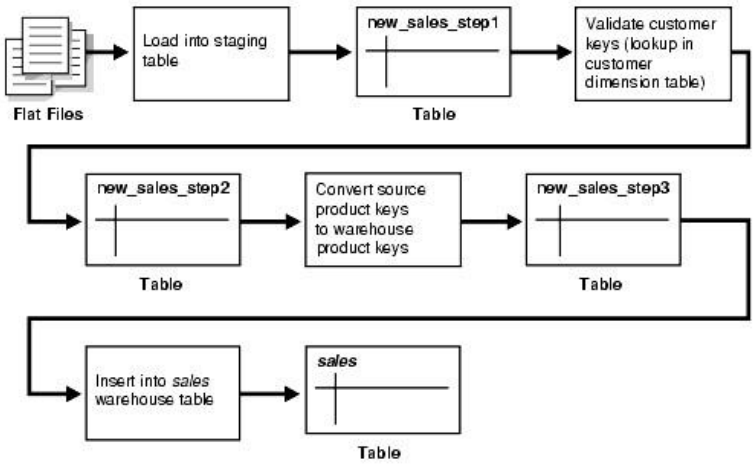
```
CREATE DIRECTORY def_dir AS
'/net/dlsun48/private/hbaer/WORK/FEATURES/et';
DROP TABLE extract_cust;
CREATE TABLE extract_cust
ORGANIZATION EXTERNAL
(TYPE ORACLE_DATAPUMP DEFAULT DIRECTORY def_dir ACCESS
PARAMETERS
(NOBADFILE NOLOGFILE)
LOCATION ('extract_cust1.exp', 'extract_cust2.exp', 'extract_cust3.exp',
        'extract_cust4.exp'))
PARALLEL 4 REJECT LIMIT UNLIMITED AS
SELECT  c.*, co.country_name, co.country_subregion, co.country_region
FROM customers c, countries co where co.country_id=c.country_id;
```

20. Loading dan Transformasi

Transformasi Data seringkali sangat komplek, dalam hal waktu proses, bagian proses ekstraksi, transformasi dan loading yang paling membutuhkan banyak biaya. Proses ini boleh jadi merentang dari konversi data sederhana hingga teknik pengumpulan data komplek yang ekstrim. Dari perspektif arsitektural, Data dapat ditransformasikan dengan dua cara :

- *Multistage Data Transformation*

  Logika transformasi data bagi kebanyakan Data Warehouse terdiri dari beberapa tahapan. Sebagai contoh, dalam transformasi rekod baru yang dimasukkan ke dalam sebuah tabel penjualan (*sales*), boleh jadi terdapat tahapan transformasi logik yang terpisah untuk memvalidasi masing-masing *key* dimensi. Gambaran secara grafis dari proses transformation logic adalah sbb :



Gambar Transformasi Data Multistage

- *Pipelined Data Transformation*

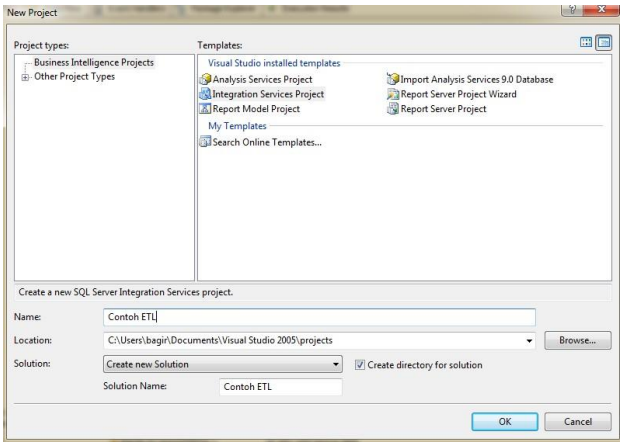
  Arus proses ETL dapat diubah secara dramatis dan database menjadi sebuah bagian integral solusi ETL. Fungsionalitas barunya melukiskan beberapa pembentukan tahapan proses penting yang kuno ketika beberapa yang lainnya dapat dimodel kembali untuk menambah arus data dan transformasi data menjadi lebih dapat diukur. Kegiatannya bergeser dari transformasi serial hingga proses load (dengan kebanyakan kegiatan dilakukan diluar database) atau load-kemudian proses transformasi untuk meningkatkan transformasi selagi loading.

21. Teknik ETL menggunakan SQL Server 2005

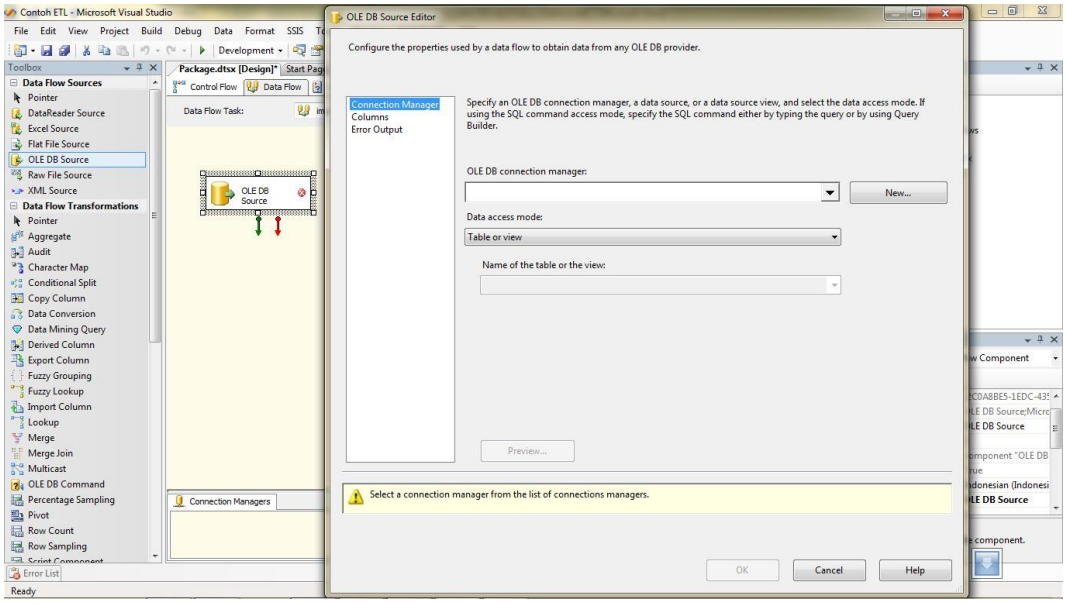
Sebagai contoh proses ETL menggunakan Aplikasi Vendor, akan kita gunakan Microsoft SQL Server 2005 untuk proses ETL dari beberapa sumber data yang berbeda seperti Flat File, Excel dan RDBMS menjadi Data Warehouse. Yang harus dipersiapkan antara lain :

i. Data dari Database
ii. Data Teks File (Flat)
iii. Data Excel

1. Buka SQL Server Business Intelligence Development Studio
2. Buat Sebuah project baru Integration Service Project dengan nama “Contoh ETL” seperti gambar dibawah ini :



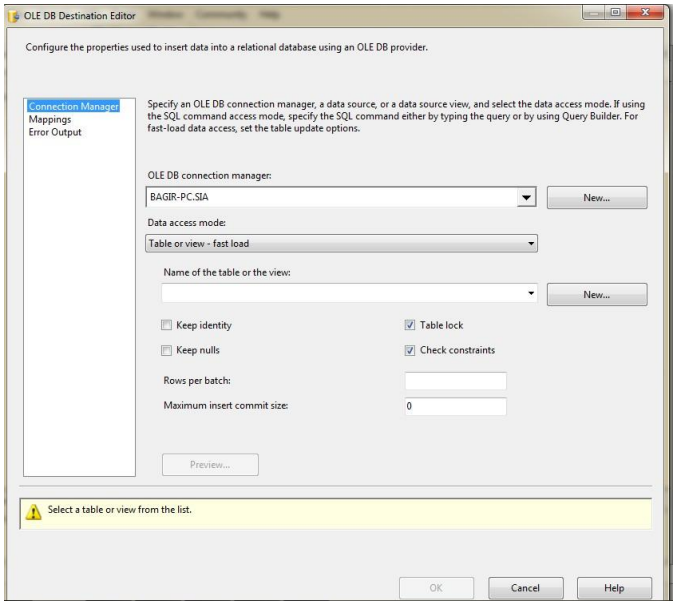
3. Klik Ok
4. Selanjutnya lanjutkan dengan tahapan Load data dari Database
   a. Drag Data Flow Task dari toolbox, ganti namanya dengan "import data registrasi"
   b. Klik 2x pada Data Flow Task, sehingga menu Data Flow tampil
   c. Drag Ole Db Source dari ToolBox, kemudian klik 2x sehingga tampilannya seperti gambar dibawah ini :



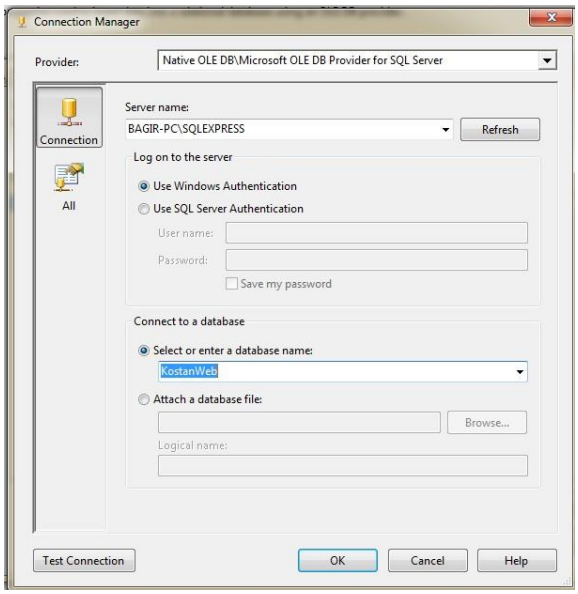
   d. Buat koneksi baru dengan memilih button "new"
   e. Pada menu configure "Ole Db Connection Manager" pilih New
   f. Pada Form Connection Manager yang tampil tentuka provide koneksi menggunakan "Microsoft Ole DB provider for SQL Server
   g. Tulis nama Server pada server Name
   h. Dan Tentukan database yang akan dikoneksikan, contoh dalam kasus ini database SIA

i. Lakukan Test Connection
j. Klik OK 2x
k. Tentukan tabel yang akan diproses, misalnya tabel registrasi
l. Pilih tab column dibawah Connection Manager, kemudian klik OK
m. Ganti nama OleDb Data Source dengan “registrasi source data”

n. Tambahkan komponen ole DB Destination
o. Ganti namanya dengan “registrasi data”
p. Gerakkan panah hijau dari “registrasi source data” ke registrasi data”
q. Klik 2x pada “registrasi data” yang sudah diubah namanya sehingga tampil menu seperti gambar dibawah ini :

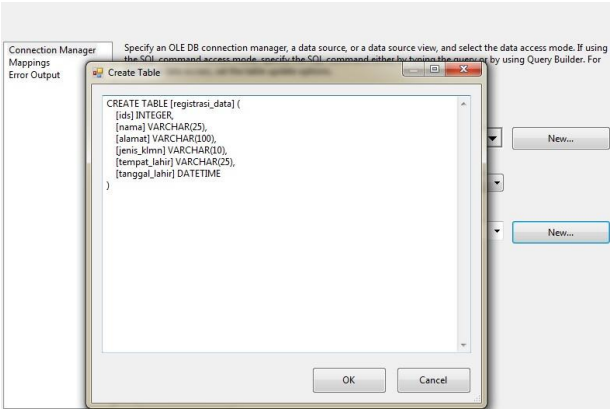


r. Klik button “new”
s. Pada configure ole DB Connection Manager klik New
t. Tentukan server target dan databasenya, misalkan dalam modul ini yang menjadi target adalah server dengan nama BAGIR-PC\SQLEXPRES, database KostanWeb seperti gambar



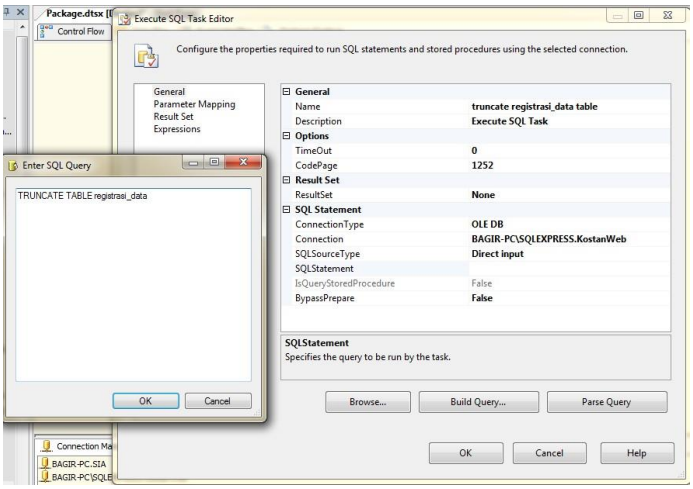
u. Klik Ok 2x

v. Klik New pada Button New sehingga tampilannnya seperti gambar dibawah ini :



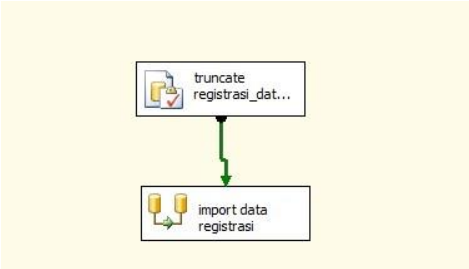
w. Klik OK, lalu pilih tab Mapping, Klik OK

x. Kembali ke Tab Control Flow, Drag Execute SQL Task ke Grid, ganti nama dengan "Truncate registrasi_data table", klik 2x, sehingga tampil menu seperti gambar dibawah ini :



y. Klik OK, dan klik Button “Parse Query”

z. Klik OK, dan simpan pekerjaan



22. Load Data dari File Teks

a. Drag For Each Container dari toolbox, dan ganti namanya dengan “loop propinsi”

b. Tambahkan Data Flow Task ke dalam loop tersebut

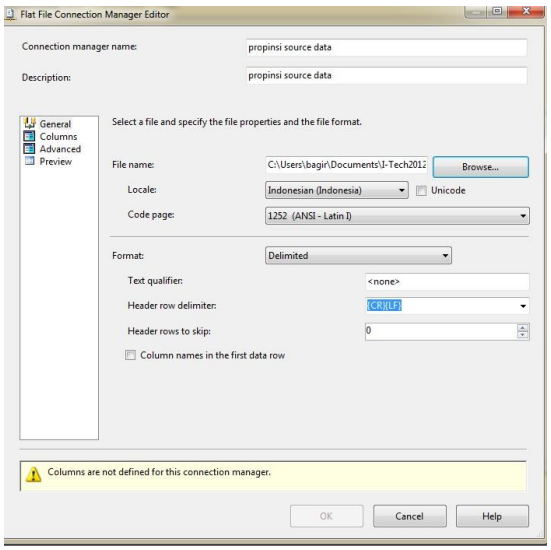
c. Ganti properti Name dengan “Import Loop

d. Klik Data Flow tersebut 2x sehingga masuk ke Data Flow

e. Drag Flat File Source dan Ole Db Destination dari ToolBox

f. Klik 2x pada Flat File Source

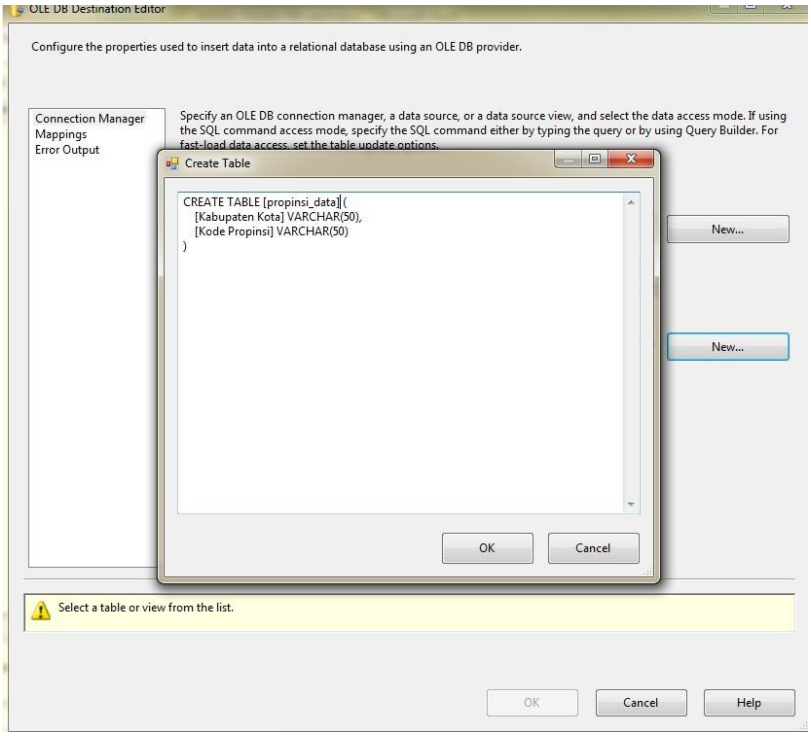
g. Pada form yang tampil, klik new, beri nama Connection dan browse file yang ditentukan. Sehingga hasilnya tampak seperti gambar



h. Beri tanda check pada colum names in the first data row, kemudian pilih tab columns. Klik OB, pilih tab Columns dan klik OK lagi

i. Klik 2x pada Ole DB Destination, dengan tampilan seperti gambar dibawah ini :

j. Klik New untuk membuat tabel baru, ganti kueri sehingga seperti gambar form dibawah ini :



k. Klik OK, pilih Tab Mapping, lihat hasilnya, Klik OK

l. Ubah nama Flat File Source dan Ole DB Destination masing-masing dengan "Propinsi Source Data" dan "Propinsi Data"

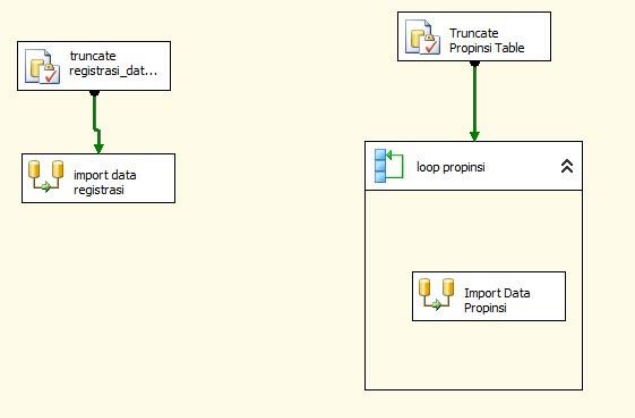
m. Kembali ke tab Control Flow,

n. Tambah Execute SQL Task, ganti namanya dengan "Truncate Propinsi Table"

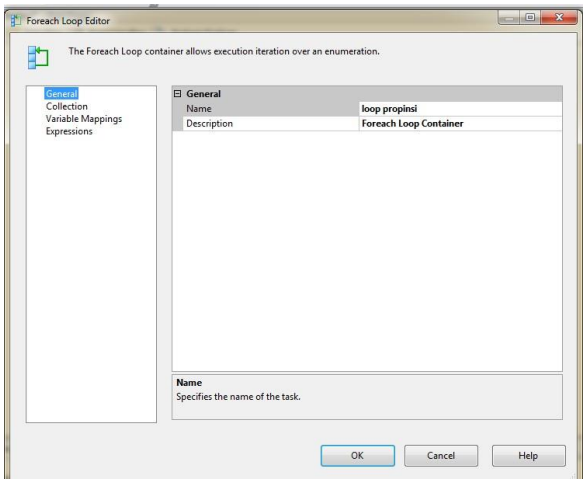
o. Klik 2x dan tentukan koneksinya, tuliskan kuerinya seperti gambar dibawah ini:

p. Klik OK

q. Klik Button "Parse Query" dan OK

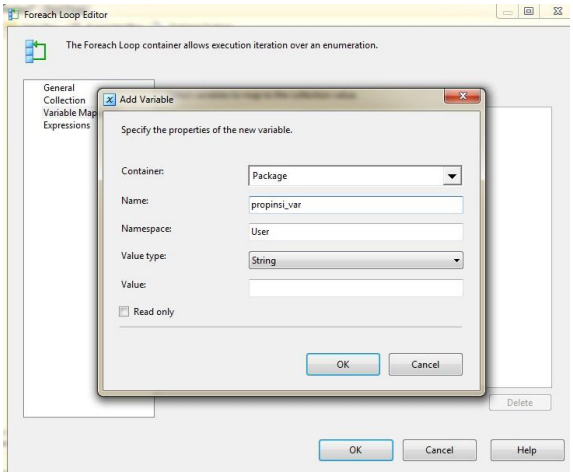


r. Klik Loop Propinsi 2x



s. Pilih Tab Collection, tentukan folder Enumerator Configuration dan ekstension file *.txt, kemudian pilih tab Variable Maping

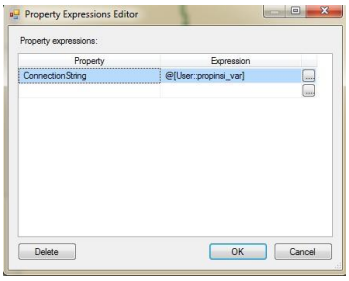
t. Pilih New Variable shingga tampil form seperti gambar :



u. Klik OK 2x

v. Kemudian Klik Propinsi Source data pada bagian paling bawah

w. Masuk ke properties, pilih expression, tentukan properties dengan "ConnectionString" dan Expression dengan "propinsi_var" seperti gambar dibawah ini :



x. Klik OK dan Save Pekerjaan dengan memilih Save All

23. Load Data dari File Excel

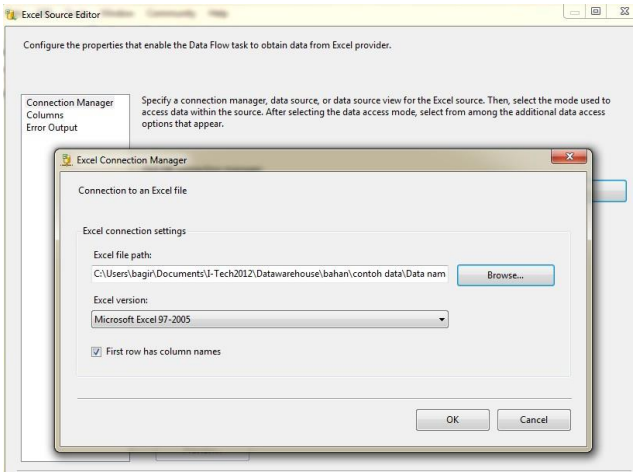
a. Drag Data Flows Task dari Toolbox ke Halaman Desain Control Flow

b. Ganti namanya dengan "import data siswa", klik 2x

c. Pada halaman Data Flow Desain yang tampil, drag Excel Data Source dan Ole Db Destination

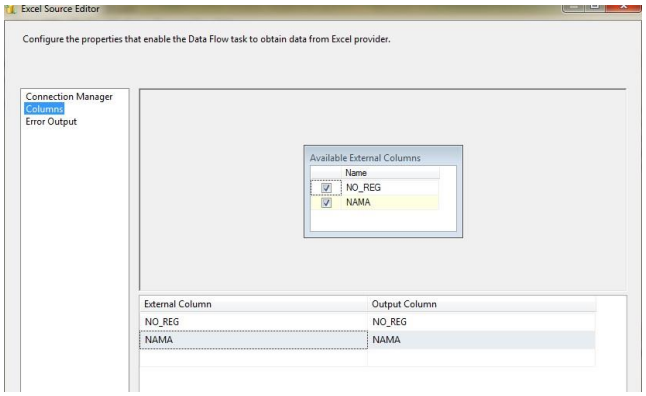
d. Klik 2x Excel Data Source sehingga tampil Form Excel Source Editor,

e. Klik New, dan browse sumber Data Excel, lalu klik OK



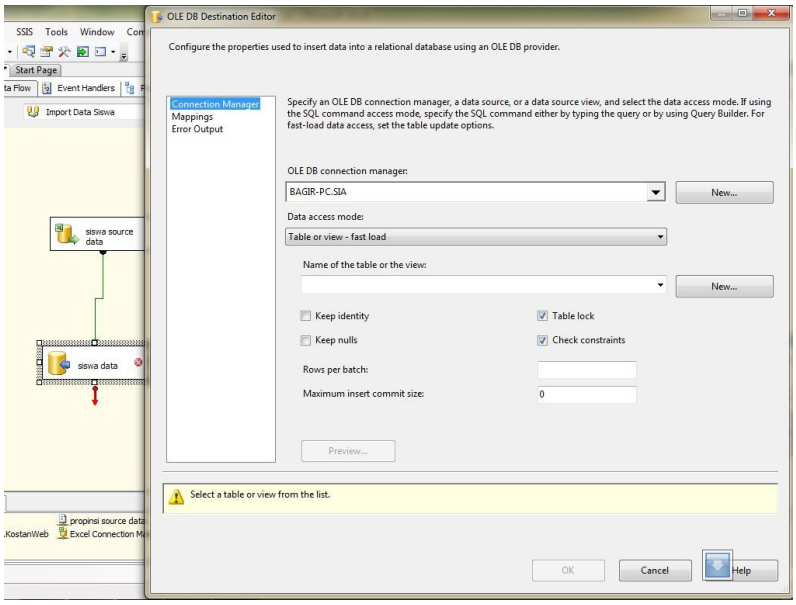
f. Tentukan Name Of Excel Sheet.

g. Pilih Tab Column sehingga hasilnya seperti gambar dibawah ini :

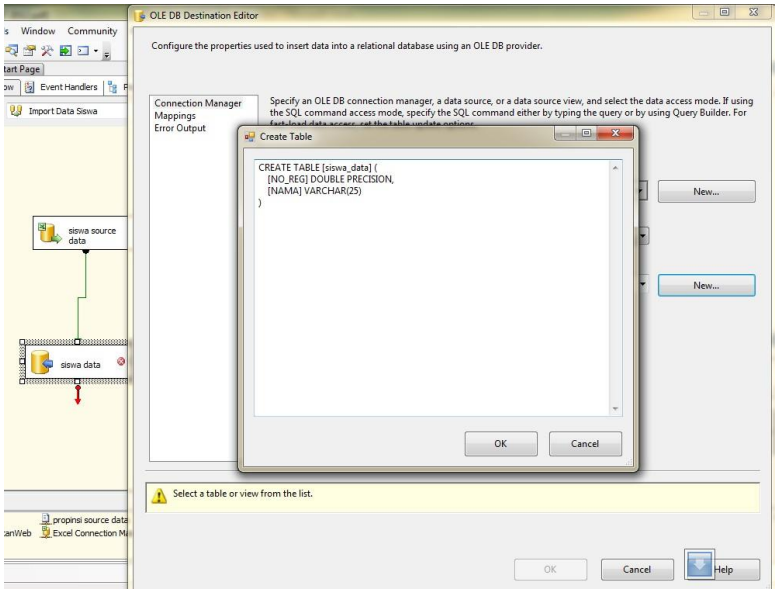


h. Klik OK.

i. Klik 2x pada siswa data, sehingga tampilannya seperti gambar dibawah ini :



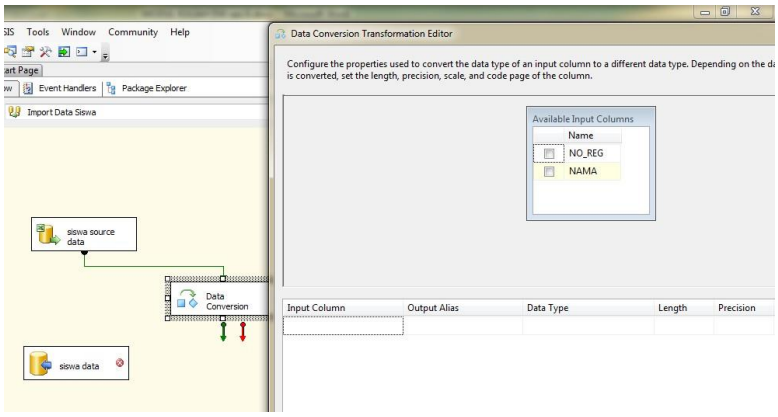
j. Klik New sehingga tampak seperti gambar dibawah ini :



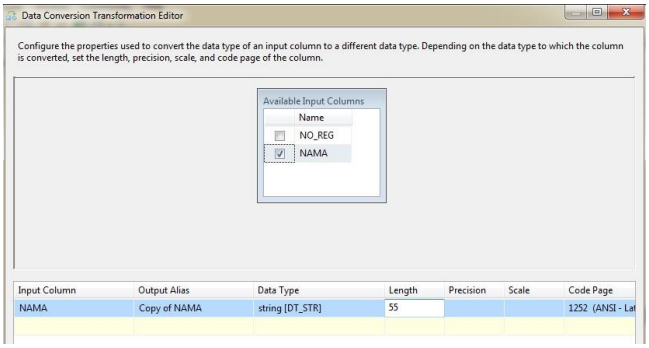
k. Klik OK, 2x

l. Dari hasil tampak bahwa kolom tidak bisa dikonversikan langsung, maka disini dibutuhkan transformasi

m. Hapus link antara kedua objek diatas, dan Drag Data Conversion dari ToolBox, Klik 2x pada Data Conversion setelah memberikan link ke Data Conversion dari data source



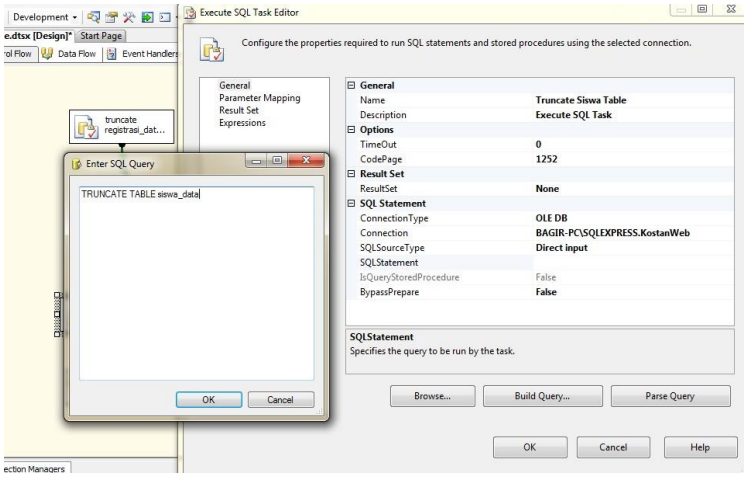
n. Lakukan hal seperti gambar dibawah ini :



o. Lanjutkan dengan OK dan Klik 2x pada Destination Siswa Data
Lakukan perubahan seperti gambar dibawah ini:

p. Klik OK dan kembali ke Control Flow

q. Drag Execute SQL Task dari ToolBox, ganti nama dengan Truncate Siswa Table, klik 2x, lalu lakukan modifikasi sehingga hasilnya seperti gambar dibawah ini:



r. Klik OK, lakukan Parse Query dan Klik OK

s. Hubungan Execute SQL dengan Data Flow Task

t. Simpan Pekerjaan

u. Dan Uji coba dengan RUN.

Content

- What and Why OLAP
- OLAP Applications
- OLAP Benefits
- OLAP Key Features
- Representation of Multi-dimensional Data
- OLAP Tools – Features
- OLAP Tools – Categories
- Multi-dimensional OLAP (MOLAP)
- Relational OLAP (ROLAP)
- Hybrid OLAP (HOLAP)
- Desktop OLAP (DOLAP)

What is OLAP

- OLAP is the dynamic synthesis, analysis, and consolidation of large volumes of multi-dimensional data.
- OLAP is the term that describes a technology that uses multi-dimensional view of aggregate data to provide quick access to strategic information for the purposes of advanced analysis.
- OLAP enables users to gain a deeper understanding and knowledge about various aspects of their corporate data through fast, consistent, interactive access to a variety of possible views of data.

- While OLAP systems can easily answer ‘who?’ and ‘what?’ questions, it is easier ability to answer ‘what if?’ and ‘why?’ type questions that distinguishes them from general- purpose query tools.
- The types of analysis available from OLAP range from basic navigation and browsing (referred to as ‘slicing’ and dicing’) , to calculations, to more complex analysis such as time series and complex modeling.

OLAP APPLICATION

- Finance: Budgeting, activity-based costing, financial performance analysis, and financial modeling.
- Sales: Sales analysis and sales forecasting.

- Marketing: Market research analysis, sales forecasting, promotions analysis, customer analysis, and market/customer segmentation.
- Manufacturing: Production planning and defect analysis.

OLAP KEY FEATURE

- Multi-dimensional views of data.
- Support for complex calculations.
- Time Intelligence.

OLAP BENEFIT

- Increased productivity of business end-users, IT developers, and consequently the entire organization.
- Reduced backlog of applications development for IT staff by making end-users self- sufficient enough to make their own schema changes and build their own models.
- Retention of organizational control over the integrity of corporate data as OLAP applications are dependent on data warehouses and OLTP systems to refresh their source data level.
- Reduced query drag and network traffic on OLTP systems or on the data warehouse.
- Improved potential revenue and profitability by enabling the organization to respond more quickly to market demands.

Representation of Multi-Dimensional Data

- OLAP database servers use multi-dimensional structures to store data and relationships between data.
- Multi-dimensional structures are best-visualized as cubes of data, and cubes within cubes of data. Each side of a cube is a dimension.

City	Time	Total Revenue
Glasgow	Q1	29726
Glasgow	Q2	30443
Glasgow	Q3	30582
Glasgow	Q4	31390
London	Q1	43555
London	Q2	48244
London	Q3	56222
London	Q4	45632
Aberdeen	Q1	53210
Aberdeen	Q2	34567
Aberdeen	Q3	45677
Aberdeen	Q4	50056
.....	.....	.....
.....	.....	.....

(a)

Time \ City	Glasgow	London	Aberdeen	.....
Q1	29726	43555	53210	.....
Q2	30443	48244	34567	.....
Q3	30582	56222	45677	.....
Q4	31390	45632	50056	.....

(b)

Representation of Multi-Dimensional Data

- Multi-dimensional databases are a compact and easy-to-understand way of visualizing and manipulating data elements that have many inter-relationships.
- The cube can be expanded to include another dimension, for example, the number of sales staff in each city.
- The response time of a multi-dimensional query depends on how many cells have to be added on-the-fly.
- As the number of dimensions increases, the number of cube's cells increases exponentially.

- Multi-dimensional OLAP supports common analytical operations, such as:
  - Consolidation: involves the aggregation of data such as 'roll-ups' or complex expressions involving interrelated data. Foe example, branch offices can be rolled up to cities and rolled up to countries.
  - Drill-Down: is the reverse of consolidation and involves displaying the detailed data that comprises the consolidated data.

- Slicing and dicing: refers to the ability to look at the data from different viewpoints. Slicing and dicing is often performed along a time axis in order to analyze trends and find patterns.

OLAP Tools – Features

- In 1993, E.F. Codd formulated twelve rules as the basis for selecting OLAP tools:
  - Multi-dimensional conceptual view
  - Transparency
  - Accessibility
  - Consistent reporting performance
  - Client-server architecture
  - Generic dimensionality
  - Dynamic sparse matrix handling
  - Multi-user support
  - Unrestricted cross-dimensional operations
  - Intuitive data manipulation
  - Flexible reporting
  - Unlimited dimensions and aggregation levels

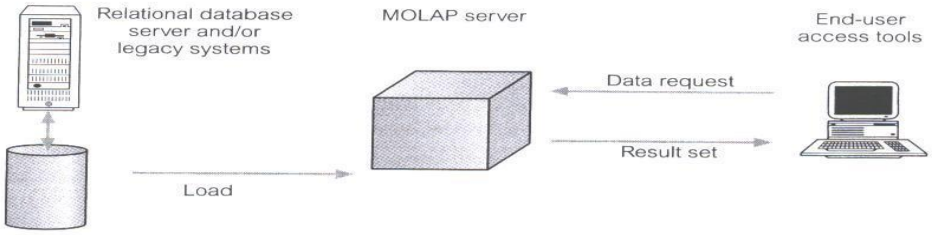
OLAP Tools – Categories

- OLAP tools are categorized according to the architecture used to store and process multi-dimensional data.
- There are four main categories of OLAP tools as defined by Berson and Smith (1997) and Pends and Greeth (2001) including:
  - Multi-dimensional OLAP (MOLAP)
  - Relational OLAP (ROLAP)
  - Hybrid OLAP (HOLAP)
  - Desktop OLAP (DOLAP)

Multi-dimensional OLAP (MOLAP)

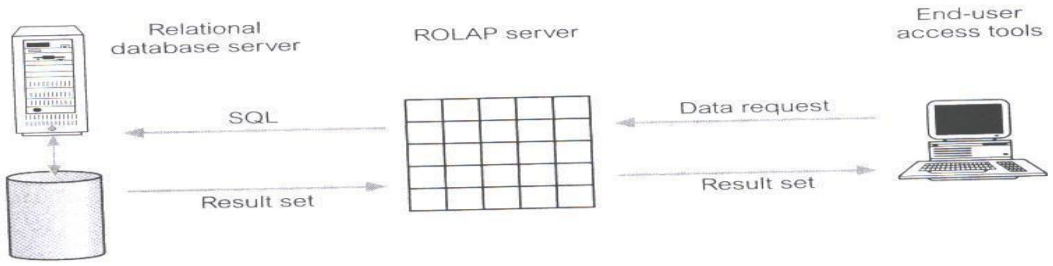
- MOLAP tools use specialized data structures and multi-dimensional database management systems (MDDBMS) to organize, navigate, and analyze data.
- To enhance query performance the data is typically aggregated and stored according to predicted usage.

- MOLAP data structures use array technology and efficient storage techniques that minimize the disk space requirements through sparse data management.
- The development issues associated with MOLAP:
  - Only a limited amount of data can be efficiently stored and analyzed.
  - Navigation and analysis of data are limited because the data is designed according to previously determined requirements.
  - MOLAP products require a different set of skills and tools to build and maintain the database.



Relational OLAP (ROLAP)

- ROLAP is the fastest-growing type of OLAP tools.
- ROLAP supports RDBMS products through the use of a metadata layer, thus avoiding the requirement to create a static multi-dimensional data structure.
- This facilitates the creation of multiple multi-dimensional views of the two-dimensional relation.
- To improve performance, some ROLAP products have enhanced SQL engines to support the complexity of multi-dimensional analysis, while others recommend, or require, the use of highly denormalized database designs such as the star schema.
- The development issues associated with ROLAP technology:
  - Performance problems associated with the processing of complex queries that require multiple passes through the relational data.
  - Development of middleware to facilitate the development of multi-dimensional applications.
  - Development of an option to create persistent multi-dimensional structures, together with facilities o assist in the administration of these structures.

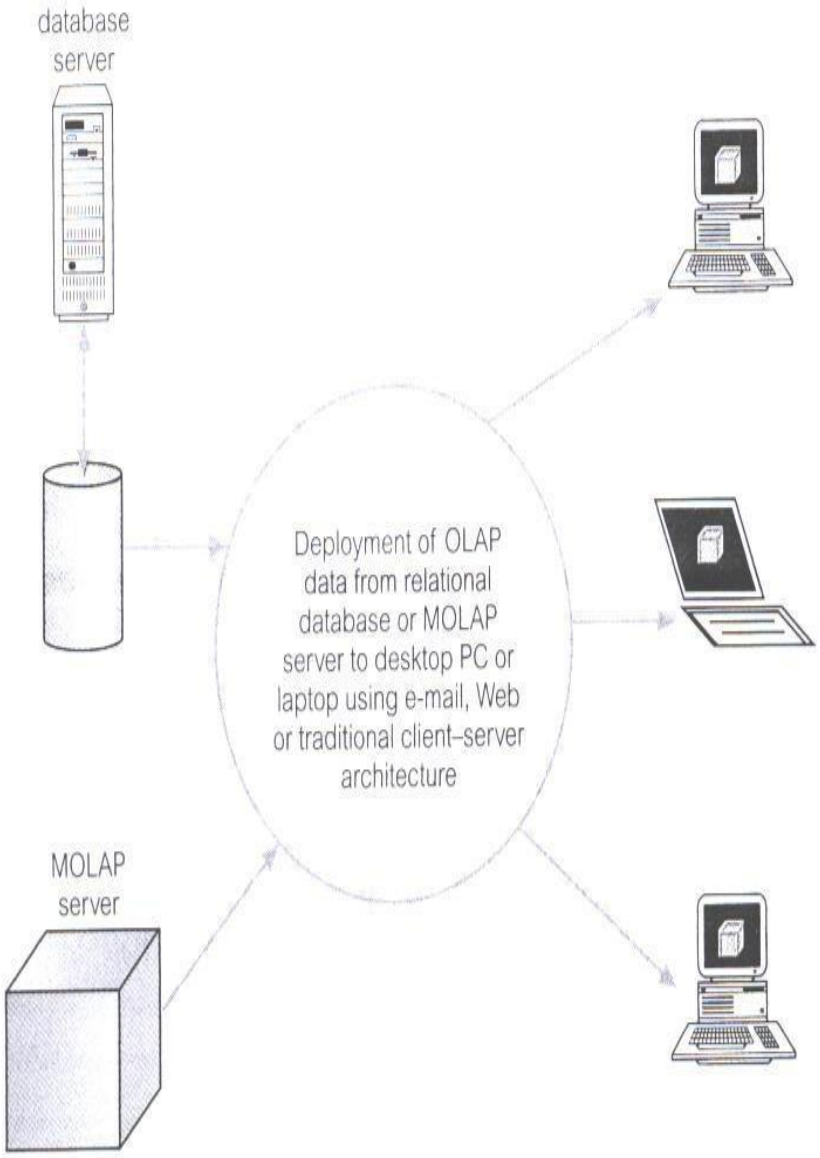


Hybrid OLAP (HOLAP)

- HOLAP tools provide limited analysis capability, either directly against RDBMS products, or by using an intermediate MOLAP server.
- HOLAP tools deliver selected data directly from DBMS or via MOLAP server to the desktop (or local server) in the form of data cube, where it is stored, analyzed, and maintained locally is the fastest-growing type of OLAP tools.
- The issues associated with HOLAP tools:
  - The architecture results in significant data redundancy and may cause problems for networks that support many users.
  - Ability of each user to build a custom data cube may cause a lack of data consistency among users.
  - Only a limited amount of data can be efficiently maintained.

Desktop OLAP (DOLAP)

- DOLAP tools store the OLAP data in client-based files and support multi-dimensional processig using a client multi-dimensional engine. DOLAP requires that relatively small eextracts of data are held on client machines. This data may be distributed in advance or on demand (possibly through the Web).
- The administration of a DOLAP database is typically performed by a central server or processing routine that prepares data cubes or sets of data for each user.
- The development issues associated with DOLAP are as follows:
  - Provision of appropriate security controls to support all parts of the DOLAP environment.
  - Reduction in the effort involved in deploying and maintaining the DOLAP tools.
  - Current trends are towards thin client machines.


database
server
Deployment of OLAP
data from relational
database or MOLAP
server to desktop PC or
laptop using e-mail, Web
or traditional client–server
architecture
MOLAP
server

Pada bab ini akan dipelajari tentang ;

1. Kebutuhan dari suatu basis data terdistribusi
2. Perbedaan antara sistem basis data terdistribusi , pemrosesan terdistribusi, dan sistem basis data paralel
3. Keuntungan dan kerugian yang dimiliki oleh DDBMS
4. Masalah keragaman pada DDBMS
5. Konsep dasar dari jaringan
6. Fungsi- fungsi yang harus di lengkapi oleh DDBMS
7. Arsitektur dari DDBMS
8. Masalah utama yang berhubungan dengan perancangan basis data terdistribusi , penamaan fragmentasi , replikasi dan alokasi data
9. Bagaimana melakukan suatu fragmentasi
10. Tingkatan transparansi pada DDBMS
11. Perbandingan kriteria untuk DDBMS

PENDAHULAN

Motivasi utama di belakang pengembangan sistem basis data adalah suatu keinginan untuk menyatukan data operasional dari suatu organisasi dan pengaksesan data yang terkontrol. Integrasi data dan kontrol data telah diimplementasikan pada bentuk data tersentralisasi, namun hal ini bukan merupakan tujuan dari pengembangan sistem basis data. Adanya perkembangan pada jaringan komputer menghasilkan suatu bentuk desentralsasi . Pendekatan desentralisasi ini merupakan gambaran dari suatu organisasi yang memiliki banyak cabang organisasi, dimana terbagi – bagi menjadi beberapa divisi, departemen, proyek dan masih banyak lagi, dan dalam bentuk infrastruktur dan akan terbagi – bagi kembali menjadi beberapa kantor cabang, pabrik-pabrik dimana setiap unit tersebut mengoperasionalkan datanya secara sendiri – sendiri. (Date,2000). Data yang digunakan secara bersama- sama dan efisiensi dalam pengaksesan data harus diiringi dengan perkembangan dari sistem basis data terdistribusi, yang merupakan

refleksi dari struktur organisasi, sehingga data dapat diakses dimana saja dan melakukan penyimpanan data di lokasi yang memang data tersebut sering digunakan.

Distribusi DBMS harusnya dapat mengatasi sekumpulan permasalahan informasi (islands of information). Basis data terkadang dianggap sebagai kumpulan elektronik saja yang terbatas dan tidak dapat di akses, seperti daerah yang terpencil. Dan DDBMS merupakan jawaban dari masalah geografi, masalah arsitektur komputer , masalah protokol komunikasi dan lain-lainnya.

KONSEP

Untuk membahas mengenai DBMS terdistribusi , terlebih dahulu mengetahui apa yang di maksud dengan basis data terdistribusi dan DBMS terdistribusi.

Basis data terdistribusi ; Secara logik keterhubungan dari kumpulan-kumpulan data yang digunakan bersama-sama, dan didistribusikan melalui suatu jaringan komputer.

DBMS Terdistribusi ; ***Sebuah sistem perangkat lunak yang mengatur basis data terdistribusi dan membuat pendistribusian data secara transparan.***

DDBMS memiliki satu logikal basis data yang dibagi ke dalam beberapa fragment. Dimana setiap fragment disimpan pada satu atau lebih komputer dibawah kontrol dari DBMS yang terpisah , dengan mengkoneksi komputer menggunakan jaringan komunikasi.

Masing- masing site memiliki kemampuan untuk mengakses permintaan pengguna pada data lokal dan juga mampu untuk memproses data yang disimpan pada komputer lain yang terhubung dengan jaringan.

Pengguna mengakses basis data terdistribusi dengan menggunakan dua aplikasi yaitu aplikasi lokal dan aplikasi global, sehingga DDBMS memiliki karakteristik yaitu :

- Kumpulan dari data logik yang digunakan bersama-sama
- Data di bagi menjadi beberapa fragment

- Fragment mungkin mempunyai copy (replika)
- Fragment / replika nya di alokasikan pada yang digunakan
- Setiap site berhubungan dengan jaringan komunikasi
- Data pada masing-masing site dibawah pengawasan DBMS
- DBMS pada masing-masing site dapat mengatasi aplikasi lokal, secara otonomi
- Masing-masing DBMS berpastisipasi paling tidak satu global aplikasi.

Dari definisi tersebut , sistem diharapkan membuat suatu distribusi yang transparan. Basis data terdistribusi terbagi menjadi beberapa fragment yang disimpan di beberapa komputer dan mungkin di replikasi, dan alokasi penyimpanan tidak diketahui pengguna . Adanya Transparansi di dalam basis data terdistribusi agar terlihat sistem ini seperti basis data tersentralisasi. Hal Ini mengacu pada prinsip dasar dari DBMS (Date,1987b). Transparansi memberikan fungsional yang baik untuk pengguna tetapi sayangnya mengakibatkan banyak permasalahan yang timbul dan harus diatasi oleh DDBMS.

Point utama dari definisi basis data terdistribusi adalah sistem terdiri dari data yang secara fisik di distribusikan pada beberapa site yang terhubung dengan jaringan.

Jika data nya tersentralisasi walaupun ada pengguna lain yang mengakses data melewati jaringan , hal ini bukan disebut dengan DDBMS melainkan pemrosesan secara distribusi.

Paralel DBMSs

DDBMS memiliki perbedaan dengan paralel DBMS.

Paralel DBMSs ; Sistem manajemen basis data ini menggunakan beberapa prosesor dan disk yang dirancang untuk dijalankan secara paralel , apabila di mungkinkan, selama hal tersebut digunakan untuk memperbaiki kinerja dari DBMS

Sistem DBMS berbasis pada sistem prosesor tunggal dimana sistem prosesor tunggal tidak memiliki kemampuan untuk berkembang, untuk menghitung skala efektifitas dan biaya, keandalan dan kinerja dari sistem. Paralel DBMS di jalankan oleh berbagai multi prosesor . Paralel DBMS menghubungkan beberapa mesin yang berukuran kecil untuk menghasilkan keluaran sebuah mesin yang berukuran besar dengan skalabilitas yang lebih besar dan keandalan dari basis datanya.

Untuk menopang beberapa prosesor dengan akses yang sama pada satu basis data, DBMS paralel harus menyediakan manajemen sumber daya yang dapat diakses bersama. Sumber daya apa yang dapat digunakan bersama, dan bagaimana sumber daya tersebut di implementasikan, mempunyai efek langsung pada kinerja dan skalabilitas dari sistem , hal ini tergantung dari aplikasi atau lingkungan yang digunakan.

Ada tiga arsitektur yang digunakan pada paralel DBMS yaitu :

a. Penggunaan memory bersama (*share memory*)
b. Penggunaan disk bersama (*share disk*)
c. Penggunaan secara sendiri-sendiri (*share nothing*)

Arsitektur pada penggunaan secara sendiri – sendiri (*share nothing*) hampir sama dengan DBMS terdistribusi, namun pendistribusian data pada paralel DBMS hanya berbasis pada kinerja nya saja. Node pada DDBMS adalah merupakan pendistribusian secara geographic, administrasi yang terpisah , dan jaringan komunikasi yang lambat, sedangkan node pada paralel DBMS adalah hubungan dengan komputer yang sama atau site yang sama.

Penggunaan Memori Bersama (*Share Memory*) adalah sebuah arsitektur yang menghubungkan beberapa prosesor di dalam sistem tunggal yang menggunakan memori secara bersama – sama (gbr 1.3). Dikenal dengan SMP (Symmetric Multiprocessing), metode ini sering digunakan dalam bentuk workstation personal yang mensupport beberapa mikroprosesor dalam paralel dbms, RISC (Reduced Instruction Set Computer) yang besar berbasis mesin sampai bentuk mainframe yang besar. Arsitektur ini menghasilkan pengaksesan data yang sangat cepat

yang dibatasi olehbeberapa prosesor , tetapi tidak dapat digunakan untuk 64 prosesor dimana jaringan komunikasi menjadi masalah (terjadinya bottleneck).

Penggunaan Disk Bersama (*Share Disk*) adalah sebuah arsitektur yang mengoptimalkan jalannya suatu aplikasi yang tersentrallisasi dan membutuhkan keberadaan data dan kinerja yang tinggi (Gbr 1.4). Setiap prosesor dapat mengakses langsung semua disk , tetapi prosesor tersebut memiliki memorinya sendiri – sendiri. Seperti halnya penggunaan secara sendiri – sendiri arsitektur ini menghapus masalah pada penggunaan memori bersama tanpa harus mengetahui sebuah basis data di
partisi. Arsitektur ini di kenal dengan **cluster**

Penggunaan Secara sendiri – sendiri (*Share nothing*) ; sering di kenal dengan Massively parallel processing (MPP) yaitu arsitektur dari beberapa prosesor di mana setiap prosesor adalah bagian dari sistem yang lengkap , yang memiliki memori dan disk (Gbr 1.5). Basis data ini di partisi untuk semua disk pada masing – masing sistem yang berhubungan dengan basis data dan data di berikan secara transparan untuk semua pengguna yang menggunakan sistem . Arsitektur ini lebih dapat di hitung skalabilitasnya dibandingkan dengan share memory dan dapat dengan mudah mensupport prosesor yang berukuran besar. Kinerja dapat optimal jika data di simpan di lokal dbms.

Paralel teknologi ini biasanya digunakan untuk basis data yang berukuran sangat besar (terabites) atau sistem yang memproses ribuan transaksi perdetik. Paralel DBMS dapat menggunakan arsitektur yang diinginkan untuk memperbaiki kinerja yang kompleks untuk mengeksekusi kueri dengan menggunakan paralel scan, join dan teknik sort yang memperbolehkan node dari banyak prosesor untuk menggunakan bersama pemrosesan kerja yang di gunakan.

KEUNTUNGAN DAN KERUGIAN DARI DDBMS

Data dan aplikasi terdistribusi mempunyai kelebihan di bandingkan dengan sistem sentralisasi basis data. Sayangnya , DDBMS ini juga memiliki kelemahan.

KEUNTUNGAN

Merefleksikan pada bentuk dari struktur organisasinya Ada suatu organisasi yang memiliki sub organisasi di lokasi yang tersebar di beberapa tempat,.sehingga basis data yang digun akan pun tersebar sesuai lokasi dari sub organisasi berada. Penggunaan bersama dan lokal otonomi Distribusi secara geografis dari sebuah organisasi dapat terlihat dari data terdistribusinya, pengguna pada masing-masing site dapat mengakses data yang disimpan pada site yang lain. Data dapat dialokasikan dekat dengan pengguna yang biasa menggunakannya pada sebuah site, sehingga pengguna

mempunyai kontrol terhadap data dan mereka dapat secara konsekuen memperbaharui dan memiliki kebijakkan untuk data tersebut. DBA global mempunyai tanggung jawab untuk semua sistem. Umumnya sebagian dari tanggung jawab tersebut di serahkan kepada tingkat lokal, sehingga DBA lokal dapat mengatur lokal DBMS secara otonomi. Keberadaan data yang ditingkatkan Pada DBMS yang tersentralisasi kegagalan pada suatu site akan mematikan seluruh operasional DBMS. Namun pada DDBMS kegagalan pada salah satu site, atau kegagalan pada hubungan komunikasi dapat membuat beberapa site tidak dapat di akses, tetapi tidak membuat operasional DBMS tidak dapat dijalankan. Keandalan yang ditingkatkan Sebuah basis data dapat di replikasi ke dalam beberapa fragmen sehingga keberadaanya dapat di simpan di beberapa lokasi juga. Jika terjadi kegagalan dalam pengaksesan data pada suatu site di karenakan jaringan komunikasi terputus maka site yang in gin mengakses data tersebut dapat mengakses pada site yang tidak mengalami kerusakan. Kinerja yang ditingkatkan Sebuah data ditempatkan pada suatu site dimana data tersebut banyak di akses oleh pengguna, dan hal ini mempunyai dampak yang baik untuk paralel DBMS yaitu memiliki kecepatan dalam pengkasesan data yang lebih baik dibandingkan dengan basis data tersentralisasi Selanjutnya, sejak masing-masing site hanya menangani sebagian dari seluruh basis data , mengakibakan perbedaan pada pelayanan CPU dan I/O seperti yang di karakteristikan pada DBMS tersentralisasi.

Ekonomi

Grosch's Law menyatakan daya listrik dari sebuah komputer di hitung menurut biaya yang dihabiskan dari pengguna an peralatannya, tiga kali biaya peralatan, 9 kali nya dari daya listrik . Sehingga lebih murah jika membuat sebuah sistem yang terdiri dari beberapa mini komputer yang mempunyai daya yang sama jika dibandingkan dengan memiliki satu buah super komputer. Oleh karena itu lebih efektif untuk menambah beberapa workstation untuk sebuah jaringan dibandingkan dengan memperbaharui sistem mainframe. Potensi yang juga menekan biaya yaitu menginstall aplikasi dan menyimpan basis data yang diperlukan secara geograf i sehingga mempermudah operasional pada setiap situs.

Perkembangan modular

Di dalam lingkungan terdistribusi, lebih mudah untuk menangani ekspansi . Site yang baru dapat di tambahkan ke suatu jaringan tanpa mempengaruhi operational dari site - site yang ada. Penambahan ukuran basis data dapat di tangani dengan menambahkan pemrosesan dan daya tampung penyimpanan pada suatu jaringan. Pada DBMS yang tersentralisasi perkembangan akan di ikuti dengan mengubah perangkat keras dan perangkat lunak.

KERUGIAN

Pada distribusi DBMS yang digunakan adalah replikasinya, DBMS yang asli tidak digunakan untuk operasional, hal ini untuk menjaga reliabilitas dari suatu data. Karena yang digunakan replikasinya maka hal ini menimbulkan berbagai macam masalah yang sangat kompleks dimana DBA harus dapat menyediakan pengaksesa n dengan cepat , keandalan dan keberadaan dari basis data yang up to date

. Jika aplikasi di dalam DBMS yang digunakan tidak dapat menangani hal - hal tersebut maka akan terjadi penurunan pada tingkat kinerja

, keandalan dan kerberadaan dari DBMS tersebut, sehingga keuntungan dari DDBMS tidak akan terjadi.

Biaya

Meningkatnya kekompleksan pada suatu DDBMS berarti biaya untuk perawatan dari DDBMS akan lebih besar dibandingkan dengan DBMS yang tersentralisasi, seperti biaya untuk membuat jaringannya, biaya komunikasi yang berjalan , orang-orang yang ahli dalam penggunaan, pengaturan dan pengawasan dari DDBMS.

Keamanan

Pada DBMS yang tersentralisasi, pengaksesan data lebih terkontrol. Sedangkan pada DDBMS bukan hanya replikasi data yang harus di kontrol tetapi jaringan juga harus dapat di kontrol keamanannya.

Pengontrolan Integritas lebih sulit

Kesatuan basis data yang mengacu pada keabsahan dan kekonsistenan dari data yang disimpan. Kesatuan biasanya di ekspresikan p ada batasan, dimana berisi aturan untuk basis data yang tidak boleh diubah. Membuat batasan untuk integrity, umumnya memerlukan pengaksesan ke sejumlah data yang sangat besar untuk mendefinisikan batasan tersebut, namun hal ini tidak termasuk di dalam operasional update itu sendiri. Dalam DDBMS , komunikasi dan biaya pemrosesan yang dibutuhkan untuk membuat suatu batasan integrity mungkin tidak diperbolehkan.

HOMOGEN DAN HETEROGEN DDBMS

Sebuah DDBMS dapat di klasifikasikan menjadi homogen dan heterogen. Dalam sistem yang homogen, semua site menggunakan product DBMS yang sama. Dalam sistem heterogen , product DBMS yang digunakan tidak sama, begitu juga dengan model datanya sehingga sistem dapat

terdiri dari beberapa model data seperti relasional, jaringan, hirarki dan obyek oriented DBMS. Sistem homogen lebih mudah di rancang dan di atur. Pendekatan ini memberikan perkembangan yang baik, tidak mengalami kesulitan dalam membuat sebuah site baru pada DDBMS , dan meningkatkan kinerja dengan mengeksploitasikan kemampuan dalam pemrosesan paralel di beberapa site yang berbeda.

Sistem heterogen, menghasilkan beberapa site yang individual dimana mereka mengimplementasikan basis data mereka dan penyatuan data nya di lakukan di tahap berikutnya. Pada sistem ini penterjemahan di perlukan untuk mengkomunikasikan diantara beberapa DBMS yang berbeda. Untuk menghasilkan transparansi DBMS, pengguna harus dapat menggunakan bahasa pemrograman yang digunakan oleh DBMS pada lokal site. Sistem akan mencari lokasi data dan menampilkan sesuai dengan yang diinginkan.

Data yang dibutuhkan dari site lain kemungkinan :

- Memiliki hardware yang berbeda
- Memiliki product DBMS yang berbeda
- Memiliki hardware dan produk DBMS yang berbeda

Jika hardwarenya yang berbeda tetapi produk DBMS nya sama , maka yang akan di ubah adalah kode dan panjang katanya. Jika yang berbeda produk DBMSnya maka akan lebih kompleks lagi karena yang akan di ubah adalah proses pemetaan dari struktur data dalam satu model data yang sama dengan struktur data pada model data yang lain. Sebagai contoh : relasional pada model data relasional di petakan ke dalam beberapa rekord dan set di model data jaringan . Juga diperlukan perubahan pada bahasa queri yang digunakan (Contoh pada SQL Perintah SELECT di petakan kedalam model jaringan menjadi FIND atau GET). Jika keduanya yang berbeda, maka dua tipe perubahan ini diperlukan sehingga pemrosesan menjadi lebih kompleks.

Kompleksitas lainnya adalah memiliki skema konseptual yang sama, dimana hal ini di bentuk dari penyatuan data dari skema individual pada konseptual lokal. Untuk mengatasi hal tersebut di gunakan GATEWAY , dimana metode ini di gunakan untuk mengkonversi bahasa pemrograman dan model data di setiap DBMS yang berbeda ke dalam bahasa dan model data relasional . Tetapi metode ini juga memiliki keterbatasan
, yang pertama tidak mensupport manjemen transaksi, bahkan untuk sistem yang sepasang. Dengan kata lain metode ini di antara dua buah sistem hanya merupakan penterjemah query. Sebagai contoh , sebuah sistem tidak dapat mengkoordinasikan kontrol konkurensi dan transaksi pemulihan data yang melibatkan pengupdatean pada basis data yang berhubungan. Kedua, metode ini hanya dapat mengatasi masalah penterjemahan query yang di tampilkan dalam satu bahasa ke bahasa lainnya yang sama.

GAMBARAN SEBUAH JARINGAN

Jaringan (Networking) adalah kumpulan dari komputer - komputer yang terhubung dengan suatu garis komunikasi yang digunakan untuk menukar informasi.

Jaringan komputer mungkin di klasifikasikan dalam beberapa jenis. Salah satu klasifikasinya adalah menurut jarak yang digunakan untuk menghubungkan beberapa komputer : Jarak pendek (Local Area Network) atau jarak jauh (Wide Area Network)

. Sebuah Local area network (LAN) digunakan untuk menghubungkan komputer pada suatu site yang sama. Wide area network (WAN) digunakan untuk menghubungkan komputer yang jarak nya lebih jauh. Jenis lain dari Wan yaitu Metropolitan area network (MAN) yang biasanya meliputi sebuah kota atau pinggiran kota . Dengan jarak geografi yang luas , hubungan komunikasi pada WAN relatif lebih lambat dan kurang dapat diandalkan dibandingkan dengan LAN. Kecepatan pengiriman data pada WAN biasanya berkisar 33.6 kilobit per detik (dial up dengan modem) sampai 45 megabit per detik (T3 tanpa melalui saluran pribadi). Kecepatan pengiriman data pada LAN lebih tinggi yaitu 10 megabit per detik (dengan ethernet) sampai 2500 megabit per detik (ATM) dan memiliki keandalan data yang baik . Yang jelas DDBMS yang menggunakan LAN untuk komunikasi akan memberikan waktu respon yang lebih cepat dibandingkan dengan WAN.

Jika di perhatikan cara dari memilih path atau routine, dapat diklasifikasikan jaringan nya dengan point to point atau dengan broadcast. Dalam jaringan point to point, jika sebuah site ingin mengirimkan pesan ke semua site, pesan tersebut harus di pisah – pisahkan ke dalam beberapa pesan. Di jaringan broadcast , semua site mendapatkan semua pesan , tetapi masing –masing pesan memiliki awalan yang menjadi identitas site tujuan sehingga site yang lainnya di abaikan. WAN biasanya menggunakan jenis jaringan point to point dan LAN menggunakan jenis jaringan broadcast. Ringkasan mengenai jenis karakteristik dari WAN dan LAN di berikan pada tabel 1.1

WAN	LAN
Jarak dapat mencapai ribuan kilometer	Jarak dapat mencapai hingga beberapa kilometer
Hubungan komputer berjauhan	Hubungan komputer yaitu bekerjasama dalam aplikasi terdistribusi
Jaringan diatur oleh organisasi bebas (menggunakan penghubungan satelit atau line telepon)	Jaringan di atur oleh pemakai sendiri (menggunakan kabel sendiri)
Kecepatan data sekitar 33.6 Kbit /detik (saluran dengan menggunakan modem) sampai 45 mbit / detik (T3)	Kecepatan data mencapai 2500 mbit / detik (ATM)
Protokol rumit	Protokol sederhana
Routing point to point	Routing broadcast
Topologi yang digunakan tidak tentu	Menggunakan topologi BUS atau RING
Tingkat kesalahan $1:10^5$	Tingkat kesalahan $1:10^9$

Tabel 1.1
Ringkasan Karakteristik dari WAN dan LAN

Organisasi internasional untuk standarisasi telah menetapkan sebuah protokol yang mengatur cara agar sebuah sistem dapat berkomunikasi (ISO,1981) . Pendekatan yang dilakukan adalah dengan membagi jaringan dalam beberapa jenis lapisan. Protokol tersebut di kenal dengan ISO Open Systems Interconnection Model (OSI Model) , yang terdiri dari tujuh pabrikan lapisan independen. Lapisan ini mentransmisi bit yang belum di olah melewati jaringan , mengatur keterhubungan dan memastikan hubungannya bebas dari kesalahan , pengaturan rute atau lintasannya dan kontrol jaringannya, mengatur masalah antara sistem mesin yang berbeda .

PROTOKOL JARINGAN

Protokol jaringan adalah sekumpulan aturan – aturan yang menentukan bagaimana pesan antar komputer dapat terkirim , diterjemahkan dan di proses. Pada bagian ini diuraikan beberapa gambaran protokol jaringan utama.

TCP/IP (Transmission Control Protocol / Internet Protocol)

Ini adalah protokol standard komunikasi dalam internet, sekumpulan jaringan komputer di seluruh dunia. TCP memiliki tanggung jawab untuk memeriksa pengiriman data yang benar dari client ke server. IP menyediakan mekanisme routing, berdasarkan pada empat byte alamat tujuan (alamat IP). Bagian depan dari alamat IP menunjukan bagian jaringan dari alamat dan bagian belakang menunjukan bagian host dari alamat . Batas pemisah jaringan dengan bagian host dari alamat IP tidak ditentukan . TCP/IP adalah protokol terskema , yaitu semua pesan tidak hanya berisikan alamat dari pos yang di tuju tetapi juga alamat dari jaringan yang dituju . Hal ini mengijinkan pesan TCP/IP di kirim ke banyak jaringan dalam suatu organisasi atau seluruh dunia.

SPX/IPX (Sequenced Packet Exchange / Internetwork Package Exchange)

Novell membuat SPX/IPX sebagai bagian dari sistem operasi netware. Hampir sama dengan TCP, SPX menjamin bahwa pesan yang masuk sampai dengan lengkap tetapi menggunakan protokol IPX Netware sebagai mekanisme pengirimannya. Seperti IP , IPX menangani rute paket yang melewati jaringan . Tidak seperti IP, IPX menggunakan 80 bit untuk alamat, dengan 32 bit bagian alamat jaringan dan 48 bit bagian alamat host(hal ini lebih besar dibandingkan dengan yang digunakan pada IP yaitu 32 bit) IPX tidak menangani paket fragmentasi . Bagaimanapun juga salah satu yang terbaik dari IPX adalah pemberian alamat host yang otomatis. Pemakai dapat memindahkan lokasi jaringan ke tempat yang lain dan melanjutkan pekerjaan dengan mudah dengan menyambungkannya lagi ke jaringan . Ini sangat penting sekali untuk pemakai yang sering berpindah – pindah. Sampai netware 5.0 , SPX/IPX adalah protokol yang digunakan , tetapi untuk menggambarkan betapa pentingnya internet, Netware 5.0 mengangkat TCP/IP sebagai protokol yang digunakan .

NetBIOS (Network Basic Input Output System)

Protokol jaringan dikembangkan pada tahun 1984 oleh IBM dan Sytek sebagai aplikasi standard komunikasi untuk PC. Pada awalnya NetBIOS dan NetBEUI (NetBIOS dengan pengembangan tampilan pemakai) telah mempertimbangkan satu protokol . Kemudian NetBIOS banyak digunakan sejak digunakan bersama protokol NetBEUI,TCP/IP, dan SPX/IPX. NetBEUI adalah protokol jaringan yang kecil, cepat dan efisien yang disalurkan bersama produk jaringan microsoft . Bagaimanapun , ini bukan rute skema, jadi konfigurasi khusus dengan menggunakan Net BEUI untuk komunikasi bersama sebuah Lan dan TCP/IP melebihi LAN.

APPC (Advanced Program to Program Communciation)

Protokol komunikasi tingkat tinggi dari IBM yangmenyediakan sebuah program untuk berinteraksi dengan jaringan lain. Ini dapat mendukung client – server dan memperhitungkan pendistribusian dengan menyediakan pemrograman tampilan biasa pada sebuah platform IBM. Ini di dukung perintah untuk mengatur pembahasan, pengiriman, dan penerimaan data dan manajemen transaksi menggunakan dua tahap pelaksanaannya. Perangkat lunak APPC adalah salah satu bagian atau yangtersedia secara bebas, dalam semua sistem operasi non IBM lainnya. Sejak APPC hanya di dkukung oleh sistem arsitektur jaringan IBM dengan

memanfaatkan protokol LU 6.2 untuk membahas pendirian APPC dan LU 6.0 sering kali sama.

Pada bagian ini akan d bahas bagaimana efek dari distribusi suatu basis data untuk fungsi dan pembuatan aristektur DDBMS.

FUNGSI

Dalam bahasan ini, diharapkan pada DDBMS mempunyai paling tidak satu dari fungsional suatu DBMS tersentralisasi. Fungsi – fungsi pada DDBMS yaitu :

1. Memberikan pelayanan komunikasi untuk memberikan akses terhadap site- site yang terhubung baik yang site yang jarak dekat maupun yang letak nya cukup jauh dan mengijinkan pencarian data ke site – site yang terhubung.
2. Memiliki sistem katalog untuk menyimpan kumpulan detail data yang telah didstribusikan.
3. Mendistribusikan proses pencarian, termasuk optimasisasi dan pengaksesan dari jarak jauh.
4. Memberikan pengendalian keamanan untuk akses ataupun otoritas yang telah diberikan .
5. Memberikan kontrol konkurensi untuk memelihara data yang telah di replikasi.
6. Memberikan pelayanan recoveri untuk mengambil laporan yang rusak dari setiap site dan kegagalan dalam hubungan komunikasi

Pada ANSI-SPARC ada tiga tingkatan arsitektur dalam DBMS yang dimana arsitektur ini memberikan konstribusi yang banyak untuk arsitektur DDBMS. Perbedaan yang dimiliki oleh DDBMS lebih kompleks / rumit jika dibandingkan dengan arsitektur DBMS. Seperti yang dapat dilihat pada gambar 1.6 yang berisi beberapa tingkatan pada arsitektur DDBMS :

*. Kumpulan tingkatan eksternal global

*. Tingkatan global konseptual

*. Tingkatan fragmentasi dan tingkatan distribusi

*. Kumpulan tingkatan untuk masing – masing DBMS

lokal yang disesuaikan dengan arsitektur pada ANSI-

Garis dalam gambar tersebut menggambarkan pemetaan antara tingkatan – tingkatan yang

cocok dengan tingkat konseptual dalam arsitektur ANSI-SPARC.

Skema Fragmentasi dan Pendistribusian

Skema ini adalah gambaran tentang bagaimana data secara logika di pisah – pisah. Alokasi dari tingkatan ini adalah gambaran tentang ke mana data tersebut akan di si
mpan dan membuat laporan dari semua penggandaan.

Skema Lokal

Setiap DBMS lokal memiliki skemanya masing - masing . Konseptual lokal dan skema internal pembentukannya sama dengan arsitektur DBMS. Skema pemetaan memetakan fragment – fragment ke dalam alokasi skema kemudian menjadi obyek eksternal pada basis data lokal. Hal ini merupakan kemandirian dari suatu basis data dan merupakan dasar untuk mendukung keanekaragaman suatu DBMS.

ARSITEKTUR FEDERATED DBMS

Sistem ini berbeda dengan DDBMS dalam tingkat penyediaan otonomi lokalnya. Hal itu dapat di lihat dari penggambaran arsitekturnya pada gambar 1.7 , dimana pada FDBMS berbentuk *tightly coupled* dimana pada arsitektur ini terdapat skema global konseptual (SGC) yang merupakan subset dari lokal konseptual skema berisi data

dari setiap lokal sistem yang dapat digunakan bersama . GCS dari sistem *tightly coupled* mempunyai kesatuan data dari setiap skema konseptual dan eksternal nya. Sedangkan pada DDBMS, SGC adalah gabungan dari semua skema konseptual pada setiap lokal sistem.

FDBMS diperdebatkan tidak memiliki skema global konseptual (Liwtin,1988) yang mana sistem ini lebih condong kepada *loosely coupled* dimana skema eksternal terdiri dari satu atau lebih skema konseptual.

KOMPONEN ARSITEKTUR DDBMS

Pada arsitektur DDBMS terdapat empat komponen utama yaitu :

1. Komponen DBMS lokal

2. Komponen Komunikasi Data (DC)
3. Katalog Sistem Global (GCS)
4. Komponen DDBMS Terdistribusi

Keempat komponen ini dapat di lihat dari gambar 1.8

Komponen Lokal DBMS

Komponen LDBMS ini adalah komponen standard dari DBMS, yang memiliki tanggung jawab untuk mengontrol data lokal pada masing – masing lokasi yang telah memiliki basisdata. Hal ini berarti setiap lokasi memiliki SGC masing – masing yang berisi semua informasi tentang data . Pada sistem homogen komponen LDBMS memiliki produk sistem yang sama yang di replikasi di setiap lokasi. Dan pada sistem heterogen akan ada dua lokasi dengan produk DBMS yang berbeda atau bentuk DBMSnya.

Komponen Komunikasi Data

Komponen ini adalah perangkat lunak dan perangkat keras yang memungkinkan semua lokasi dapat berkomunikasi dengan baik satu sama lain. Komponen komunikasi data berisikan informasi tentang site dan jaringannya.

Katalog Sistem Global (GCS)

GCS memiliki kesamaan fungsi dengan sistem katalog pada tersentralisasi. GCS menangani informasi yang spesifik mengenai pendistribusian dari suatu sistem, seperti fragmentasi, penggandaan dan alokasi nya. Komponen ini dapat mengatur dirinya sendiri seperti mendistribusikan basisdata dan fragmentasi , replikasi keseluruhan atau sentralisasi. Pada GCS yang melakukan replikasi secara keseluruhan menjamin otonomi dari setiap site , seperti melakukan modifikasi harus di beritahukan kepada seluruh site yang terhubung. GCS yang tersentalisasi juga menjanjikan otonomi untuk sitenya dan sangat sensitif terhadap suatu kesalahan pada suatu sitenya.

Pendekatan ini digunakan pada sistem terdistribusi R^* (Williams at al,1982). Dalam sistem ini terdapat katalog lokal di setiap site yang terdiri dari meta data yang berhubungan data yang disimpan. Untuk Keterhubungannya disimpan di beberapa site, hal ini merupakan tanggung jawab pada setiap lokal katalog untuk mencatat definisi dari setiap fragmen dan setiap replikas dari setiap fragmen dan mencatat dimana fragment atau replika tersebut di alokasikan. Kapanpun fragmen atau replika di gunakan pada lokasi yang berbeda, lokal katalog harus selalu mengupdate perubahan tersebut, sehingga fragmen atau replika dapat diandalkan keberadaannya.

Komponen DBSM Terdistribusi

Komponen DDBMS adalah pengendalian unit di semua sistem.

PERANCANGAN RELASIONAL BASIS DATA TERDISTRIBUSI

Faktor - faktor yang dianjurkan untuk digunakan pada basis data terdistribusi yaitu :

1. Fragmentasi : Sebuah relasi yang terbagi menjadi beberapa sub-sub relasi yang disebut dengan fragment, sehingga disebut juga distribusi. Ada dua buah fragmentasi yaitu horisontal dan vertikal. Horisontal fragmentasi yaitu subset dari tupel sedangkan vertikal fragmentasi subset dari atribut.
2. Alokasi, setiap fragmen disimpan pada situs dengan distribusi yang optimal.
3. Replikasi, DDBMS dapat membuat suatu copy dari fragmen pada beberapa situs yang berbeda.

Definisi dan alokasi dari fragmen harus berdasarkan pada bagaimana basis data tersebut digunakan.

Perancangan harus berdasarkan kuantitatif dan kualitatif informasi. Kuantitatif informasi digunakan pada alokasi data sedangkan kualitatif informasi digunakan untuk fragmentasi.

Kuantitatif informasi termasuk :

- Seberapa sering aplikasi di jalankan
- Situs mana yang aplikasinya dijalankan
- Kriteria kinerja untuk transaksi dan aplikasi

Kualitatif informasi termasuk transaksi yang dieksekusi pada aplikasi, termasuk pengaksesan relasi, atribut dan tuple , tipe pengaksesan(R atau W) dan predikat dari operasional.
Definisi dan alokasi dari fragment menggunakan strategi untuk mencapai obyektifitas yang diinginkan :

1. Referensi Lokal
   Jika memungkinkan data harus disimpan dekat dengan yang menggunakan. Bila suatu fragmen digunakan di beberapa lokasi , akan menguntungkan jika fragmen data tersebut disimpan di beberapa lokasi juga.

2. Reliabilitas dan Availabilitas yang ditingkatkan

Keandalan dan ketersediaan data ditingkatkan dengan replikasi. Ada salinan lain yang disimpan di lokasi yang lain.

3. Kinerja yang di terima

   Alokasi yang tidak baik dapat mengakibatkan bottleneck terjadi, sehingga akan mengakibatkan banyaknya permintaan dari beberapa lokasi yang tidak dapat dilayani dan data yang diminta menjadi tidak up to date menyebabkan kinerja turun.

4. Seimbang antara kapasitas penyimpanan dan biaya

   Pertimbangan harus diberikan pada ketersediaan infrastruktur dan biaya untuk penyimpanan di setiap lokasi, sehingga untuk efisiensi dapat digunakan tempat penyimpanan yang tidak mahal.

5. Biaya komunikasi yang minimal

   Pertimbangan harus diberikan untuk biaya akses jarak jauh. Biaya akan minimal ketika kebutuhan lokal maksimal atau ketika setiap site menduplikasi data nya sendiri. Bagaimanapun ketika data yang di replikasi telah di update. Maka data yang ter-update tersebut harus di duplikasi ke seluruh site, hal ini yang menyebabkan naiknya biaya komunikasi.

ALOKASI DATA

Ada empat strategis menurut penempatan data : sentralisasi, pembagian partisi, replikasi yang lengkap dan replikasi yang dipilih.

1. Sentralisasi

Strategi ini berisi satu basis data dan DBMS yang disimpan pada sa tu situs dengan pengguna yang didistribusikan pada jaringan (pemrosesan distribusi). Referensi lokal paling rendah di semua situs, kecuali situs pusat, harus menggunakan jaringan untuk pengaksesan semua data. Hal ini berarti juga biaya komunikasi tinggi.

Keandalan dan keberadaan rendah, kesalahan pada situs pusat akan mempengaruhi semua sistem basis data.

2. Partisi (Fragmentasi)

Strategi ini mempartisi basis data yang dipisahkan ke dalam fragmen-fragmen, dimana setiap fragmen di alokasikan pada satu site. Jika data yang dilokasikan pada suatu site, dimana data tersebut sering digunakan maka referensi lokal akan meningkat. Namun tid ak akan ada replikasi , dan biaya penyimpanan nya rendah, sehingga keandalan dan keberadaannya juga rendah, walaupun pemroses an distribusi lebih baik dari pada sentralisasi. Ada satu kelebihan pada sentralisasi yaitu dalam hal kehilangan data, yang hilang hanya ad a pada site yang bersangkutan dan aslinya masih ada pada basis data pusat. Kinerja harus bagus dan biaya komunikasi r endah jika distribusi di rancang dengan sedemikian rupa..

3. Replikasi yang lengkap

Strategi ini berisi pemeliharaan salinan yang lengkap dari suatu basis data di setiap site. Dimana referensi lokal, keberadaan dan keandalan dan kinerja adalah maksimal. Bagaimanapun biaya penyimpanan dan biaya komunikasi untuk mengupdate besar sekali biayanya. Untuk mengatasi masalah ini, biasanya digunakan snapshot . Snapshot digunakan untuk menyalin data pada waktu yang telah ditentukan. Data yang disalin adalah hasil update per periode , misalkan per minggu atau perjam, sehingga data salinan tersebut tidak selalu up to date. Snapshot juga digunakan untuk mengimplementasikan table view di dalam data terdistribusi untuk memperbaiki waktu yang digunakan untuk kinerja operasional dari suatu basis data.

4. Replikasi yang selektif

Strategi yang merupakan kombinasi antara partisi,replikasi dan sentralisasi. Beberapa item data di partisi untuk mendapatkan referensi
lokal yang tinggi dan lainnya, yang digunakan di banyak lokasi dan tidak selalu di update adalah replikasi ;selain dari itu di lakukan sentralisasi. Obyektifitas dari strategi ini untuk mendapatkan semua keuntungan yang dimiliki oleh semua strategi dan bukan kelemahannya. Strategi ini biasa digunakan karena fleksibelitasnya.

FRAGMENTASI

Kenapa harus dilakukan fragmentasi ?

Ada empat alasan untuk fragmentasi :

1. Kebiasaan ; umumnya aplikasi bekerja dengan tabel views dibandingkan dengan semua hubungan data. Oleh karenanya untuk distribusi data , yang cocok digunakan adalah bekerja dengan subset dari sebuah relasi sebagai unit dari distribusi.

2. Efisien ; data disimpan dekat dengan yang menggunakan. Dengan tambahan data yang tidak sering digunakan tidak usah disimpan.

3. Paralel ; dengan fragmen-fragmen tersebut sebagai unit dari suatu distribusi , sebuah transaksi dapat di bagi kedalam beberapa sub queri yang dioperasikan pada fragmen tersebut. Hal ini meningkatkan konkurensi atau paralelisme dalam sistem, sehingga memeperbolehkan transaksi mengeksekusi secara aman dan paralel.

4. Keamanan ; data yang tidak dibutuhkan oleh aplikasi tidak disimpan dan konsukuen tidak boleh di ambil oleh pengguna yang tidak mempunyai otoritas.

Fragmentasi mempunyai dua kelemahan, seperti yang disebutkan sebelumnya :

1. Kinerja; cara kerja dari aplikasi yang membutuhkan data dari beberapa lokasi fragmen di beberapa situs akan berjalan dengan lambat.
2. Integritas; pengawasan inteegritas akan lebih sulit jika data dan fungsional ketergantungan di fragmentasi dan dilokasi pada beberapa situs yang berbeda.

Pembetulan dari fragmentasi

Fragmentasi tidak bisa di buat secara serampangan, ada tiga buah aturan yang harus dilakukan untuk pembuatan fragmentasi yaitu :

1. Kelengkapan ; jika relasi contoh R di dekomposisi ke dalam fragment R_1 , R_2 ,R_3 , … R_n , masing-masing data yang dapat ditemukan pada relasi R harus muncul paling tidak di salah satu fragmen. Aturan ini di perlukan untuk meyakinkan bahwa tidak ada data yang hilang selama fragmentasi

2. Rekonstruksi; Jika memungkinkan untuk mendefinisikan operasional relasi yang akan dibentuk kembali relasi R dari fragmen-fragmen.

   Aturan ini untuk meyakinkan bahwa fungsional ketergantungan di perbolehkan .

3. Penguraian; Jika item data d_i muncul pada fragment R_i , maka tidak boleh muncul di fragmen yang lain. Vertikal fragmentasi diperbolehkan untuk aturan yang satu ini, dimana kunci utama dari atribut harus diulanmg untuk melakukan rekonstruksi. Aturan ini untuk meminimalkan redudansi.

Tipe dari Fragmentasi

Ada dua tipe utama yang dimiliki oleh fragmentasi yaitu horisontal dan vertikal , tetapi ada juga dua tipe fragmentasi lainnya yaitu : mixed dan derived fragmentasi .

1. **Horisontal fragmentasi ;**

Fragmentasi ini merupakan relasi yang terdiri dari subset sebuah tuple . Sebuah horisontal fragmentasi di hasilkan dari menspesifikasikan predikat yang muncul dari sebuah batasan pada sebuah tuple didalam sebuah relasi. Hal ini di definisikan dengan menggunakan o perasi SELECT dari aljabar relasional . Operasi SELECT mengumpulkan tuple yang memiliki kesamaan kepunyaan; seba gai contoh, tuple yang semua nya menggunakan aplikasi yang sama atau pada situs yang sama. Berikan relasi R sebuah horisontal fragmentasi yang didef inisikan

dimana P adalah sebuah predikat yang berdasarkan atas satu atau lebih atribut dida lam suatu relasi.

Contoh : Diasumsikan hanya mempunyai dua tipe properti yaitu tipe flat dan rumah, horisontal fragmentasi dari properti untuk di sewa dari tipe properti dapat di peroleh sebagai berikut :

P_1 : σ_{tipe} = 'Rumah'(properti sewa)

P_2; σ_{tipe} = 'Flat'(properti sewa)

Hasil dari operasi tersebut akan memiliki dua fragmentasi , yang satu terdiri dari tipe yang mempunyai nilai 'Rumah' dan yang satunya yang mempunyai nilai "Flat'.

Fragment P_1

Pno	Street	Area	City	Pcode	Type	Rooms	Rent	Cno	Sno	Bno
PA14	16 Holl	Dee	Aber	AB75S	Rumah	6	650	CO46	SA9	B7
PG21	18 Dell	Hynd	Glas	G12	Rumah	4	500	CO87	SG37	B3

Fragment P_2

Pno	Street	Area	City	Pcode	Type	Rooms	Rent	Cno	Sno	Bno
PL94	6 Arg	Dee	Aber	AB74S	Flat	4	450	CO67	SL41	B5
PG4	8 Law	Hynd	Glas	G50	Flat	4	400	CO70	SG14	B3
PG16	2 Man	Part	Glas	G67	Flat	3	300	CO90	SG14	B3

Fragmentasi seperti ini mempunyai keuntungan jika terjadi transaksi pada beberapa aplikasi yang berbeda dengan Flat ataupun Rumah.

Fragmentasi skema memuaskan aturan pembetulan (Correctness rules) :

1. Kelengkapan ; setiap tuple pada relasi muncul pada fragment
   P_1 atau
   P_2

2. Rekonstruksi ; relasi Properti sewa dapat di rekonstruksi dari fragmentasi menggunakan operasi Union , yakni :

   P_1 U P_2 = Properti sewa

3. Penguraian ; fragmen di uraikan maka tidak ada tipe properti yang mempunyai tipe flat ataupun rumah.

   Terkadang pemilihan dari strategi horisontal fragmentasi terlihat jelas. Bagaimanapun pada kasus yang lain, diperlukan penganalisaan secara detail pada aplikasi. Analisa tersebut termasuk dalam menguji predikat atau mencari kondisi yang digunakan oleh transaksi atau queri pada aplikasi. Predikat dapat berbentuk sederhana (atribut tunggal) ataupun kompleks (banyak atribut). Predikat setiap atribut mungkin mempunyai nilai tunggal ataupun nilai yang banyak. Untuk kasus selanjutnya nilai mungkin diskrit atau mempunyai range.

   Fragmentasi mencari group predikat minimal yang dapat digunakan sebagai basis dari fragmentasi skema. Set dari predikat disebut lengkap jika dan hanya

   jika ada dua tuple pada fragmen yang sama bereferensi pada kemungkinan yang sama oleh beberapa aplikasi . Sebuah predikat dinyatakan relevan jika ada paling tidak satu aplikasi yang dapat mengakses hasil dari fragment yang berbeda.

2. Vertikal Fragmentasi

Adalah relasi yang terdiri dari subset pada atribut

Fragmentasi vertikal ini mengumpulkan atribut yang digunakan oleh beberapa aplikasi. Di definisikan menggunakan operasi PROJECT pada aljabar relasional. Relasi R sebuah vertikal fragmentasi di definisikan :

$\Pi a_1,a_2,\ldots a_n (R)$

dimana $a_1,a_2,\ldots a_n$ merupakan atribut dari relasi R

contoh :Aplikasi Payroll untuk PT. Dream Home membutuhkan nomor pokok daari Staff (Sno) dan Posisi, Sex, DOB,Gaji dan NIN atribut setiap anggota dari staff tersebut; departemen kepegawaian membutuhkan ; Sno,Fname,Lname, Alamat,Tel_no dan Bno atribut, Vertikal fragmentasi dari

staff untuk contoh ini diperlukan sebagai berikut : S_1

$= \Pi_{Sno,posisi,sex,dob,gaji,nin}(Staff)$

$S_2 = \Pi_{Sno,Fname,Lname,Alamat,Tel_no,Bno}(Staff)$

Akan menghasilkan dua buah fragmen , kedua buah fragmen tersebut berisi kunci utama (Sno) untuk memberi kesempatan yang aslinya untuk di rekonstruksi. Keuntungan dari vertikal fragmentasi ini yaitu fragmen-fragmen tersebut dapat disimpan pada situs yang memerlukannya. Sebagai tambahan kinerja yang di tingkatkan, seperti fragmen yang diperkecil di bandingkan dengan yang aslinya.

Fragmentasi ini sesuai dengan skema kepuasan pada aturan pembetulan (Correcness Rules):

1. Kelengkapan ; setiap atribut di dalam relasi staff muncul pada setiap fragmen S_1 dan S_2
2. Rekonstruksi ; relasi staff dapat di rekonstruksi dari fragmen menggunakan operasi natural join , yakni :
   $S_1 \bowtie S_2 = Staff$
3. Penguraian ; fragment akan diuraikan kecuali kunci utama, karena diperlukan untuk rekonstruksi .

Fragment S_1

Sno	Posisi	Sex	DOB	Salary	NIN
SL21	Manager	M	1-oct-60	300000	WK44201B
SG37	Snr Ass	F	10-nov-65	150000	WL43251C
SG14	Deputy	M	24-mar-70	100000	WL22065B

SA9	Assistant	F	20-jan-70	90000	WM53218D

Fragment S_2

Sno	Fname	Lname	Alamat	Tel_no	Bno
SL21	John	White	19 Taylor London	0171-884-5112	B5
SG37	Ann	Beech	81 George Glasgow	0141-848-3345	B3
SG14	David	Ford	63 Ashby Glasgow	0141-339-2177	B3
SA9	Marie	Howe	2 Elm Abeerdeen		B7

3. Campuran Fragmentasi

Fragmentasi ini terdiri dari horisontal fragmentasi setelah itu vertikal fragmentasi, atau vertikal fragmentasi lalu horisontal fragmentasi.

Fragmentasi campuran ini di definisikan menggunakan operasi SELECT dan PROJECT pada aljabar relasional.

Relasi R adalah fragmentasi campuran yang didefinisikan sbb :

$\sigma_P(\pi_{a_1,a_2,\ldots a_n}(R))$ atau $\pi_{a_1,a_2,\ldots a_n}(\sigma_P(R))$

dimana p adalah predikat berdasarkan satu atau lebih atribut R dan $a_1,a_2,\ldots a_n$ adalah atribut dari R

contoh :

Vertikal fragmentasi staff dari aplikasi payroll dan departemen kepegawaian kedalam :

$S_1 = \pi_{Sno,posisi,sex,dob,gaji,nin}(Staff)$

$S_2 = \pi_{Sno,Fname,Lname,Alamat,Tel_no,Bn}(Staff)$

Lalu lakukan horisontal fragmentasi pada fragmen S_2 menurut nomor cabang: S_{21}

$= \sigma_{Bno = B3}(S2)$

$S_{22} = \sigma_{Bno = B5}(S2)$

$S_{23} = \sigma_{Bno = B7}(S2)$

Fragment S_1

Sno	Posisi	Sex	DOB	Salary	NIN
SL21	Manager	M	1-oct-60	300000	WK44201B
SG37	Snr Ass	F	10-nov-65	150000	WL43251C
SG14	Deputy	M	24-mar-70	100000	WL22065B
SA9	Assistant	F	20-jan-70	90000	WM53218D

Fragment S_{21}

Sno	Fname	Lname	Alamat	Tel_no	Bno
SG37	Ann	Beech	81 George Glasgow	0141-848-3345	B3
SG14	David	Ford	63 Ashby Glasgow	0141-339-2177	B3

Fragment S_{22}

Sno	Fname	Lname	Alamat	Tel_no	Bno
SL21	John	White	19 Taylor London	0171-884-5112	B5

Fragment S_{23}

Sno	Fname	Lname	Alamat	Tel_no	Bno
SA9	Marie	Howe	2 Elm Abeerdeen		B7

Dari fragmentasi tersebut akan menghasilkan tiga buah fragmen yang baru berdasarkan nomor cabang. Fragmentasi tersebut sesuai dengan aturan pembetulan.(Correction rules)

1. Kelengkapan ; Setiap atribut pada relasi staff muncul pada fragmentasi S_1 dan S_2 dimana setiap tupel akan mencul pada fragmen S_1 dan juga fragmen S_{21} ,S_{22} dan S_{23} .

2. Rekonstruksi ; relasi staff dapat di rekonstruksi dari fragmen menggunakan operasi Union dan Natural Join , yakni: S_1 (S_{21} U S_{22} U S_{23}) = Staff

3. Penguraian ; penguraian fragmen ; tidak akan ada Sno yang akan muncul di lebih dari satu cabang dan S_1 dan S_2 adalah hasil penguraian kecuali untuk keperluan duplikasi kunci utama.

4. Derived Horisontal Fragmentation

Beberapa aplikasi melibatkan sua atau lebih relasi gabungan. Jika relasi disimpan ditempat yang berbeda, mungkin akan memiliki perbedaan yang siginifikan di dalam proses penggabungan tersebut. Di dalam fragmentasi ini akan lebih pasti keberadaan relasi atau fragmen dari relasi di tempat yang sama.

Derived fragmen : horisontal fragmen yang berdasarkan fragmen dari relasi yang utama

Istilah anak akan muncul kepada relasi yang mengandung foreign key dan parent pada relasi yang mengandung primari key. Derived fragmentasi di jabarkan dengan menggunakan operasi semijoin dari aljabar relasional.

Misalkan relasi anak adalah R dari relasi parent adalah S, maka fragmentasi derived digambarkan sebagai berikut :

$R_I = R \longrightarrow S_f$ L □ I □
w

Dimana w adalah nomor dari fragmen horisontal yang telah digambarkan pada S
dan f adalah atribut join

Contoh :

Suatu perusahan mempunyai aplikasi yang menggabungkan relasi staff dan PropertyForRent secara bersamaan. Untuk contoh ini di asumsikan staff telah terfragmentasi secara horisontal berdasarkan nomor cabang. Jadi data yang berhubungan dengan cabang disimpan di tempat :

S_3 = □ Bno =

$B3^{(Stsff)}$ S_4 = □ Bno

= $B5^{(Staff)}$ S_5 = □

Bno = $B7^{(Staff)}$

Diasumsikan bahwa properti PG4 diatur oleh SG14. Ini seharusnya berguna untukmenyimpan data propetri yang menggunakan strategi fragmentasi sama. Ini di peroleh dengan menggunakan derived fragmentasi untuk menfragmentasi secara horisontal relasi PropertiForRent berdasarkan nomor cabang :

Menghasilkan 3 fragmen (P_3,P_4 dan P_5) . satu terdiri dari proreprti yang diatur oleh staff dengan nomor cabang B3 (P3), yang satunya terdiri dari properti yang diatur oleh staf dengan nomor cabang B5 (P5) dan yangterakhir terdiri dari properti yang diatur oleh staff dengna nomro cabang B7 (P7) . Akan mudah dilihat skema fragmentasi ini sesuai dengan peraturan fragmentasi.

Fragment P_3

Pno	Street	City	Pcode	Type	Rooms	Rent	Cno	Sno
PG4	6Law	Glas	G11	Flat	3	350	CO40	SG149
PG36	2 Mann	Glas	G32	Flat	3	375	C093	SG37
PG21	18 Dell	Glas	G12	House	4	500	CO87	SG37
PG16	5 Nov	Glas	G12X	Flat	4	450	C093	SG14

Fragment P_4

Pno	Street	City	Pcode	Type	Rooms	Rent	Cno	Sno
PL94	6 Arg	Lon	NW1	Flat	4	400	CO87	SL41

Fragment P_5

Pno	Street	City	Pcode	Type	Rooms	Rent	Cno	Sno
PA14	16Holl	Aber	AB74S	House	6	650	CO46	SA9

5. Tidak Terdapat Fragmentasi

Strategi final adalah tidak memfragmentasikan relasi. Sebagai contoh, relasi cabang hanya mengandung sejumlah update secara berkala . Daripada mencoba untuk menfragmentasikan relasi secara horisontal, misalnya nomor cabang akal lebih masuk akal lagi untuk membiarkan relasi keseluruhan dan mereplikasi relasi cabang pada setiap sisinya.

TRANSPARANSI PADA DDBMS

Definisi dari DDBMS yang telah dijelaskan pada subbab 1.1 menyatakan bahwa sistem seharusnya melakukan distribusi yang transparan kepada pengguna. Detail dari implementasi pengguna tidak perlu mengetahuinya. DDBMS menampilkan banyak level transparan. Semua transparansi berpartisipasi di semua obyek, agar dapat membuat basis data terdistribusi ini dapat sejalan dengan basis data tersentralisasi . Ada 4 macam tipe utama dari transparansi dalam DDBMS yaitu

1. Transparansi Distribusi
2. Transparansi Transaksi
3. Transparansi Kinerja
4. Transparansi DBMS

1. Transparansi Distribusi

Distribusi transparansi memperbolehkan pengguna untuk mengetahui bahwa basis data sebagi sebuah single logikal entitas. Jika suatu DDBMS memperlihatkan transparansi terdistribusinya, pengguna tidak perlu tahu mengenai fragmentasi dari datanya ataupun locasi dimana data tersebut di simpan.

Ada suatu transparansi yang memperbolehkan pengguna untuk mengetahui apakah data telah terfragmen dan di simpan suatu di lokasi, nama dari transparansi ini yaitu : Pemetaan Transparansi Lokasi (Transparancy Local Mapping).

Contoh :

S_1 = □ staffno, position,sex,DOB,salary (STAFF) ditempatkan di site 3

S_2 = □ staffno, x,DOB,salary
fname,lname,branchno se (STAFF)

S_{21} = □ Bno = B3 (S2) ditempatkan di site 3

S_{22} = □ Bno = B5 (S) ditempatkan di site 5

S_{23} = □ Bno = B7 (S) ditempatkan di site 7

Transparansi Fragmentasi

Fragmentasi adalah tingkat tertinggi dari distribusi transparansi yang di sediakan oleh DDBMS, sehingga pengguna tidak perlu tahu mengenai data yang di fragmentasikan. Akses basis data berdasarkan pada skema globalnya, sehingga pengguna tidak perlu menspesifik nama fragmen atau lokasi datanya.

Contoh :

Select fname,lname From Staff

Where position =’Manager’;

Ini adalah statement SQL yang harus di tulis pada sistem tersentralisasi.

Transparansi Lokasi

Transparansi lokasi dalam distribusi transparansi berada pada tingkat menengah . Dengan transparansi ini , user mengetahui data tersbut di fragmentasi tidak perlu mengetahui dimana lokasi dari data tersebut.

Contoh :

SELECT fname,lname FROM S_{21}

WHERE Staffno IN (SELECT Staffno FROM S_1 WHERE Position=’Manager’)

UNION

SELECT fname,lname FROM S_{22}

WHERE Staffno IN (SELECT Staffno FROM S_1 WHERE Position=’Manager’)

UNION

SELECT fname,lname FROM S_{23}

WHERE Staffno IN (SELECT Staffno FROM S_1 WHERE Position=’Manager’);

Sekarang di perlukan nama dari fragmen dalam query. Digunakan juga join (subquery) di karenakan posisi dan fname ataupun lname muncul di beberapa vertikal fragmentasi yang berbeda. Keuntungan utama dari lokasi transparansi

adalah basis data dapat secara fisik teroragnisasi tanpa harus mempengaruhi aplikasi yang mengakses basis data tersebut.

Transparansi Replikasi

Sama dengan lokasi transparansi adalah transparansi untuk menggandakan suatu data , maksudnya pengguna tidak mengetahui data telah di fragmentasi . Transparansi ini merupakan akibat dari adanya transparansi lokasi. Bagaimanapun ada kemungkinan untuk tidak memiliki transparansi lokasi tetapi mempunyai replikasi transparansi.

Transparansi Pemetaan Lokal

Ini adalah tingkatan paling rendah pada distribusi transparansi. Dengan transparansi ini , pengguna perlu menspesifikasikan nama fragmen dan lokasi dari data items.

Contoh :

SELECT fname,lname FROM S_{21} AT SITE 3

WHERE Staffno IN (SELECT Staffno FROM S_1 AT SITE 5 WHERE Position=’Manager’)

UNION

SELECT fname,lname FROM S_{22} AT SITE 5

WHERE Staffno IN (SELECT Staffno FROM S_1 AT SITE 5 WHERE Position=’Manager’)

UNION

SELECT fname,lname FROM S_{23} AT SITE 7

WHERE Staffno IN (SELECT Staffno FROM S_1 AT SITE 5 WHERE Position=’Manager’);

Pemberian Nama Transparansi

Setiap item pada basis data yangtelah didistribusikan memiliki nama yang unik. Oleh karena DDBMS memastikan tidak ada dua site yang membuat obyek basis data dengan nama yang sama. Satu solusi dari masalah iniadalah dengan membuat *server nama terpusat*, dimana alat bantu ini berisi semua nama dari sistem sehingga jika ada yang sama akan dapat terdeteksi.

Namun masalah ini memiliki kendala yaitu :

- Kurangnya kemampuan lokal otonomi
- Masalah kinerja, jika terpusat maka akan terjadi bottleneck
- Rendahnya ketersediaan, jika site pusat gagal , site yang lain tidak dapat membuat obyek basis databyang lain.

Ada solusi alternatif yaitu dengan di gunakannya '*awalan*' suatu obyek sebagai identifier lokasi yang menciptakan obyek tersebut. Sebagai contoh relasi *Branch* di buat pada site S1 sehingga obyek tersebut dapat dinamakan S1.*Branch*. Namun jika ingin mengidentifikasi setiap fragment dan setiap salinan fragment tersebut maka dapat dibuat S1.*Branch*.F3.C2

Yang mana terdapat 2 salinan dari fragmen 3 pada relasi *Branch* yang dibuat pada site S1. Namun hal ini akan mengakibatkan kehilangan data pada transparansi terdistribusi.

Pendekatan yang lain dengan menggunakan *alias* (sinonim) untuk masing – masing obyek basis data. Seperti S1.*Branch*.F3.C2 diketahui sebagai *Localbranch* yang digunakan pengguna pada site S1. DDBMS memiliki tugas untuk memetakan *alias* mejadi obyek basis data yang sesuai.

Sistem R* yang terdistribusi membedakan antara obyek *printname* nya dengan *system wide-name* nya. *Printname* adalah nama yang pengguna gunakan yang mengacu pada suatu obyek. *System wide-name* adalah identifier internal yang unik untuk obyek yang dijamin takkan pernah di ganti. *System wide-name* terdiri dari 4 bagian yaitu :

1. Creator ID – Lokasi identifier yuang unik untuk pengguna yang menciptakan obyek
2. Creator site ID – global identifier yang unik untuk site dimana obyek dibuat
3. Local name – nama yang tidak memnuhi persyaratan untuk obyek
4. Birth-site ID – identifier yang unik untuk site dimana obyek disimpan sebagai contoh, system wide-name : Merepresentasikan sebuah obyek dengan local name localBranch, diciptakan oleh pengguna Manager di London dan disimpan di site di Glasgow.

2. Transparansi Transaksi

Transparansi ini pada lingkungan DDBMS memastikan bahwa semua transaksi terdistribusi memelihara konsistensi dan integritas basis data terdistribusinya. Transaksi terdistribusi mengakses data yang disimpan lebih dari satu tempat. Setiap transaksi di bagi menjadi beberapa subtransaksi , satu untuk mengakses site yang harus diakses; sebuah subtransaksi di represenstasikan oleh sebuah ***agent/perwakilan*.**

Contoh :

Ada sebuah transaksi T yang mencetak nama dari semua staff, dengan menggunakan skema fragmentasi yang di definisikan S_1,S_2,S_{22},dan S_{23} . Substransaksi dapat didefiniskan T_{S3},T_{S5}, dan T_{S7} untuk mewakili agen yang berada di lokasi 3, 5 dan 7. Setiap subtransaksi mencetak nama – nama staff di setiap lokasi tersebut.

Time	T_{S3}	T_{S5}	T_{S7}
t_1	Begin transaction	Begin transaction	Begin transaction
t_2	Read(fname,lname)	Read(fname,lname)	Read(fname,lname)
t_3	Print (fname,lname)	Print (fname,lname)	Print (fname,lname)
t_4	End_transaction	End_transaction	End_transaction

Kesatuan dari transaksi terdistribusi merupakan dasar dari konsep transaksi, namun DDBMS harus juga menjamin kesatuan dari setiap subtransaksi. Oleh karena itu tidak hanya harus menjamin sinkronisasi dari subtransaksi dengan local transaksi lainnya yang di operasionalkan bersamaan di sebuah lokasi. Tapi juda memastikan sinkronisasi dari subtransaksi – subtransaksi dengan transaksi global yang berjalan secara serempak di lokasi yangsama maupun di lokasi yang berbeda. Transparansi transaksi di dalam sebuah DBMS terdistribusi di lengkapi oleh bagan fragmentasi, bagan pendistribusian dan bagan replikasi.

Transparansi Konkurensi

Transparansi konkurensi dimiliki oleh DDBMS jika hasil dari semua transaksi konkuren (didistribusi ataupun yang tidak didistribusi) di laksanakan secara independen atau pun dalam satu waktu dan menjamin data yang dihasilkan konsisten dan terupdate dengan benar, hal ini sesuai dengan prinsip dasar yang dimiliki oleh basis data tersentralisasi namun ada penambahan dikarenakan bentuk nya DDBMS maka harus menjamin transaksi lokal ataupun global tidak bertentangan satu sama lain. Dengan cara yang sama, DDBMS harus memastikan konsistensi dari semua subtransaksi global.

Replikasi membuat konkurensi menjadi lebih kompleks. Jika salinan dari suatu replikasi data di perbaharui , update terbaru tersebut harus secepatnya di sebarkan ke semua salinan

yang ada. Strateginya adalah menyebarkan setiap perubahan data menjadi satu kesatuan operasional data dari sebuah transaksi. Namun, jika salah satu site yang memegang salinan data tidak dapat dicapai ketika pengupdate sedang dilakukan , dikarenakan site ataupun hubungan komunikasinya sedang gagal, maka transaksi di tunda sampai site tersebut dapat dicapai. Jika terdapat banyak salinan item data, kemungkinan transaksi konkurensi akan tidak sukses. Alternatif lain untuk membatasi hal tersebut yaitu dengan melakukan pengupdate data hanya untuk site yang saat itu ada. Strategi selanjutnya memperbolehkan pengupdate-an terhadap salinan data yang tidak dilakukan secara bersamaan, terkadang setelah basis data yang aslinya terupdate. Penundaan untuk mendapatkan kembali konsistensi dari data dapat terjadi antara beberapa detik sampai dengan beberapa jam.

Transparansi Kegagalan

DBMS tersentralisasi memiliki kemampuan untuk pemulihan data yang digunakan jika terjadinya kegagalan dalam bertransaksi. Jenis kegagalan yang dimiliki oleh DBMS tersentralisasi yaitu : sistem crash, kesalahan media, kesalahan perangkat lunak, bencana alam dan sabotase. Pada DDBMS juga memiliki jenis – jenis kegagalan yaitu :

- Kehilangan data
- Kegagalan hubungan komunikasi
- Kegagalan pada site
- Partisi jaringan

DDBMS harus memastikan kesatuan dari global transaksi, artinya memastikan subtransaksi pada global transaksi semua berhasil ataupun dibatalkan. Oleh karena itu DDBMS harus menyamakan transaksi global untuk memastikan semua subtransaksi telah sukses sebelum dicatat BERHASIL / COMMIT.

Klasifikasi Transaksi

Sebelum menyelesaikan penjelasan mengenai transaksi, akan dijelaskan secara singkat mengenai klasifikasi transaksi yang telah didefinisikan pada IBM arsitektur basis data relasional terdistribusi (DRDA). Pada arsitektur ini ada empat tipe transaksi , setiap tingkatan mempunyai penambahan pada kompleksitasnya di dalam interaksi dengan DBMS

1. Permintaan akses jarak jauh
   Aplikasi di satu lokasi dapat mengirimkan permintaan (perintah (SQL) ke beberapa lokasi yang jauh untuk mengeksekusi kiriman data tersebut. Permintaan di eksekusi secara keseluruhan pada lokasi tersebut dan dapat menjadi data acuan di lokasi yang jauh tersebut.
2. Satuan kerja jarak jauh (Remote Unit of Work)
   Suatu aplikasi di satu lokasi dapat mengirimkan semua perintah SQL di dalam satuan unit kerja (transaksi) ke beberapa lokasi yang jauh untuk pelaksanaanya. Semua perintah SQL dieksekusi seluruhnya di lokasi yangjauh dan hanya menjadi data acuan di lokasi tersebut. Namun site lokal

yang memutuskan mana transaksi yang akan di commit dan mana yang akan di rollback.

3. Satu kerja distribusi

Aplikasi di satulokasi dapat mengirimkan sebagian atau seluruh permintaan (perintah (SQL) di dalam suatu transaksi ke satu atau lebih lokasi yang jauh untuk mengeksekusi kiriman data tersebut. Permintaan di eksekusi secara keseluruhan pada lokasi tersebut dan dapat menjadi data acuan di lokasi yang jauh tersebut.

4. Permintaan Terdistribusi

Suatu aplikasi di suatu lokasi dapat mengirimkan sebagian atau seluruh permintaan (perintah (SQL) di dalam suatu transaksi ke satu atau lebih lokasi yang jauh untuk mengeksekusi kiriman data tersebut. Namun, perintah SQL membutuhkan akses data dari satu atau lebih lokasi (perintah SQL perlu dapat join atau union suatu relasi / fragmen yang berada di lokasi yang berbeda)

3. TRANSPARANSI KINERJA

Transparansi ini membutuhkan DBMS untuk menjadi seperti DBMS terpusat. Di dalam lingkungan terdistribusi, suatu sistem tidak harus mengalami penurunan selama melakukan arsitektur terdistribusi, sebagai contoh munculnya jaringan. Transparansi ini membutuhkan DBMS untuk membuat strategi agar dapat menghemat biaya yang dikeluarkan untuk melakukan suatu permintaan.

Didalam suatu DBMS tersentralisasi, query processor (QP) harus mengevaluasi setiap permintaan data dan melaksanakan strategi yang optimal, yang terdiri dari suatu urutan operasional yang diperintah pada basis data. Didalam suatu lingkungan terdistribusi Distribusi query prosessor (DQP) memetakan suatu permintaan data ke dalam suatu urutan operasi yang diperintahkan pada basis data lokal . Hal ini memiliki penambahan kompleksitas untuk mengaksesnya ke dlaam perhitunganfragmentasi, replikasi dan alokasi skema. DQP harus memutuskan :
Fragmen mana yang akan diakses

- Salinan dari fragmen yang mana yang akan digunakan jika fragmen akan di replikasi
- Lokasi mana yang akan digunakan

DQP membuat suatu strategi pelaksanaan yang optimal dengan menjalankan beberapa fungsi biaya. Secara umum, biaya – biaya yang berhubungan dengan suatu permintaan terdistribusi termasuk:

- Biaya waktu akses (I/O) melibatkan pengaksesn dalam data fisik pada disk
- Biaya waktu CPU pada saat melaksanakan operasi – operasi data dalam memori utama
- Biay akomunikasi dengan transmisi data melalui jaringan.

Faktor pertama adalah satu – satunya hal yang dipertimbangkan dalam suatu sistem tersentralisasi . Pada lingkungan terdistribusi, DDBMS harus menghitung biaya komunikasi, yang paling dominan dalam WAN dengan suatu bandwitdh untuuk golongan kecil kilobyte per detik . Pada kasus seperti itu, optimasi mungkin mengabaikan I/O dan biaya CPU. Namun, LAN mempunyai bandwidth tidak mungkin mengabaikan I/O dan biaya CPU seluruhnya.

Satu pendekatan untuk optimasi query memperkecil biaya total untuk waktu yang akan terjadi di dalam pelaksanaan queri (Sacco dan Yao,1982). Sebagai pendekatan alternatif ini dapat memperkecil waktu respon queri, di dalam kasus DQP Terkadang waktu respon akan signifikan menjadi lebih kecil dari biaya waktu total.

DATES'S 12 ATURAN UNTUK DDBMS

Pada bagian terakhir ini , akan di jelaskan mengenai dua belas atuarn mengenai DDBMS (Date,1987b). Dasar dari aturan ini adalah bahwa suatu DBMS terdistribusi harus dapat seperti DBMS non distribusi terhadap pengguna. Aturan ini serupa dengan dua belas aturan CODD untuk sistem relasional .

Prinsip dasar : Suatu sistem DDBMS harus terlihat seperti DBMS non distribusi untuk penggunanya.

1. Otonomi Lokal
   Tempat dalam sistemterdistribusi sudah harus otonom. Otonomi berarti :

   a. Data lokal adalah miliki DBMS lokal dan di atur sendiri oleh DBMS Lokal
   b. Operasi lokal tetap merupakan lokal operasional
   c. Semua operasi yang telah diberikan dikontrol oleh DBMS Lokal

2. Tidak adanya campur tangan site pusat
   Semua proses pelayanan, manajemen transaksi , pendekteksian deadlock ,
   optimasi queri dan manajemen dari sistem katalog adalah tanggung jawab dari lokal DBMS, dan pusat tidak memiliki wewenang untuk melakukan hal tersebut.

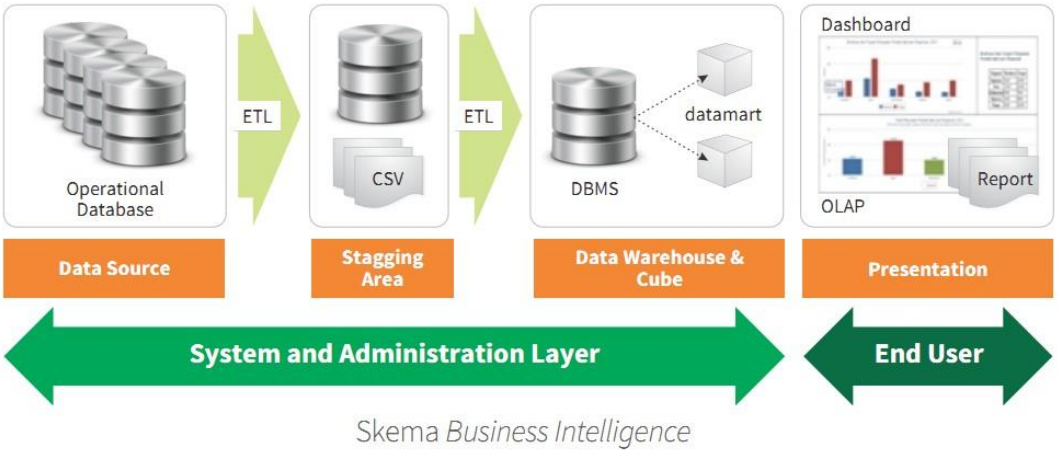
3. Operasi yang berkelanjutan
   Fungsi dari DDBMS yaitu adanya perkembangan modular dimana jika terjadi suatu ekspansi jaringan maka proses pembuatan infrastruktur tidak akan mengganggu jalannya operasional suatu data.
4. Lokasi yang mandiri
   Kebebasan lokasi sama dengan transparansi lokasi , pengguna bisa mengakses basis data dari banyak tempat. Dalam pengaksesan data tersebut semua data seolah –olah disimpan dekat dengan lokasi pengguna, bukan menjadi masalah tempat dimana data disimpan secara fisik.
5. Kebebasan Fragmentasi
   Pengguna dapat mengakses basis data tanpa harus mengetahui bagaimana data tersebut di ragmen.
6. Kebebasan replikasi
   Pengguna tidak harus mengetahui apakah data telah direplikasi atau tidak dan tidak harus mengakses suatu salinan tertentu dari item data secara langsung , juga pada saat pengguna melakukan pembaharuan data haruslah detail untuk semua data.
7. Pemrosesan query terdistribusi
   Sistem harus dapat menangani pemrosesan queri yang mereferensi ke suatu data di sejumlah site yang terhubung.
8. Pemrosesan transaksi terdistribusi
   Sistem harus mendukung sebuah transaksi sebagai sebuah unit dari suatu pemulihan data (*recovery*) . Dan menjamin bahwa global ataupun lokal transparansi harus sesuai dengan aturan ACID untuk transaksi, contohnya : penamaan, konsistensi, isolasi dan ketahanan (Automicity,Consistent, Isolation, Defence).
9. Kebebasan perangkat keras
   DDBMS harus dapat digunakan di berbagai macam platform perangkat keras.
10. Kebebasan sistem operasi
    Sesuai dengan aturan sebelumnya , maka DDBMS juga harus dapat digunakan di berbagai macam platform system operasi.
11. Kebebasan jaringan
    Sama halnya dengan aturan sebelumnya , DDBMS harus dapat digunakan di berbagai macam platform jaringan komunikasi yang berbeda.
12. Kebebasan database
    DDBMS di bentuk dari local DBMS yang berbeda, yang memungkinkan adanya model data yang berbeda. Dengan kata lain DDBMS harus dapat mendukung adanya system heterogen.

Bab 1 Business Intelligence

Business intelligence menurut wikipedia

> *refers to skills, technologies, applications and practices used to help a business acquire a better understanding of its commercial context*.

Business intelligence (BI) merujuk kepada skill, teknologi, aplikasi, dan upaya yang dilakukan untuk membantu bisnis mendapatkan pengertian yang lebih baik dari konteks komersialnya. Dalam pengertian lain, BI adalah 'sesuatu' yang dapat mengubah data menjadi keuntungan (*turn data into profits*).



Skema *Business Intelligence*

Frequently Asked Question (FAQ)

Kapan BI diperlukan?

BI diperlukan apabila eksekutif memerlukan resume dari data yang berasal dari berbagai sumber data sehingga eksekutif dapat mengambil keputusan yang tepat berdasarkan data tersebut.

Proses apa saja yang ada pada BI?

Proses yang ada pada BI antara lain extract, transform, dan load (ETL) data dari berbagai sumber data menuju data warehouse/datamart, data analisis, reporting, dan dashboarding.

Tools apa yang bisa digunakan untuk membangun BI?

Tools di pasaran sangat beragam, mulai dari yang opensource sampai yang berbayar.

Tools opensource yang direkomendasikan untuk mengimplementasikan BI?

Salah satu tools BI opensource yang direkomendasikan adalah Pentaho Community Edition. Walaupun opensource, Pentaho Community Edition dianggap telah mampu mengakomodir kebutuhan dasar dari BI.

Bab 2 Pentaho Community Edition

2.1 Pentaho

Pentaho adalah perusahaan yang mempunyai focus pada produk dan solusi business intelligence. Pentaho Community Edition atau Pentaho BI Suite adalah produk dari Pentaho yang sudah terintegrasi dengan kemampuan reporting, dashboard, data mining, workflow, dan ETL. Berikut adalah beberapa tools yang akan dipelajari pada pelatihan ini :

2.1.1 Pentaho BI Server

Produk ini adalah kumpulan dari aplikasi BI yang bersifat opensource yang berjalan di atas platform Java.

Di dalam produk inilah, reporting, dashboard dan data analysist ditampilkan setelah melalui tahapan desain.

2.1.2 Pentaho Data Integration

Pentaho Data Integration atau Kettle adalah tools yang memiliki kemampuan extract, transform, dan load (ETL) pada multi platform database. Script dari disain dapat disimpan dalam bentuk file ataupun repository. Selain itu, pada tools ini terdapat cukup banyak ‘steps’ untuk mengatur workflow control (JOB), dan data workflow (Transformation).

2.1.3 Pentaho Analysis/Mondrian OLAP Engine

Tools ini mempunyai fungsi untuk melakukan OLAP, eksekusi MDX query, star schema, dan sebagainya. Selain itu, tampilan datanya didukung dengan fungsi drill down dan charting dari hasil query.

2.1.4 Pentaho Report Designer

Tools berbasis desktop ini berperan sebagai desain report yang hasilnya dapat dikirimkan ke Pentaho BI Server. Dengan tools ini, kita bisa mendesain report sesuai dengan kebutuhan.

2.1.5 Pentaho CDF (Community Dashboard Framework)

CDF adalah framework untuk membangun dashboard di atas Pentaho BI Server.

2.1.6 Plugin Saiku for Data Analysis

Saiku adalah plugin yang dapat diinstal di Pentaho BI Server yang berguna untuk analisis data. Saiku mempunyai kelebihan dibandingkan dengan data analisis bawaan pentaho, yaitu lebih user friendly dalam penggunaannya dan lebih baik tampilannya.

Bab 3 Pentaho BI Server

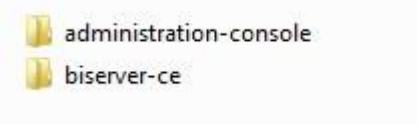
3.1 Instalasi

Berikut langkah-langkah instalasi Pentaho BI Server

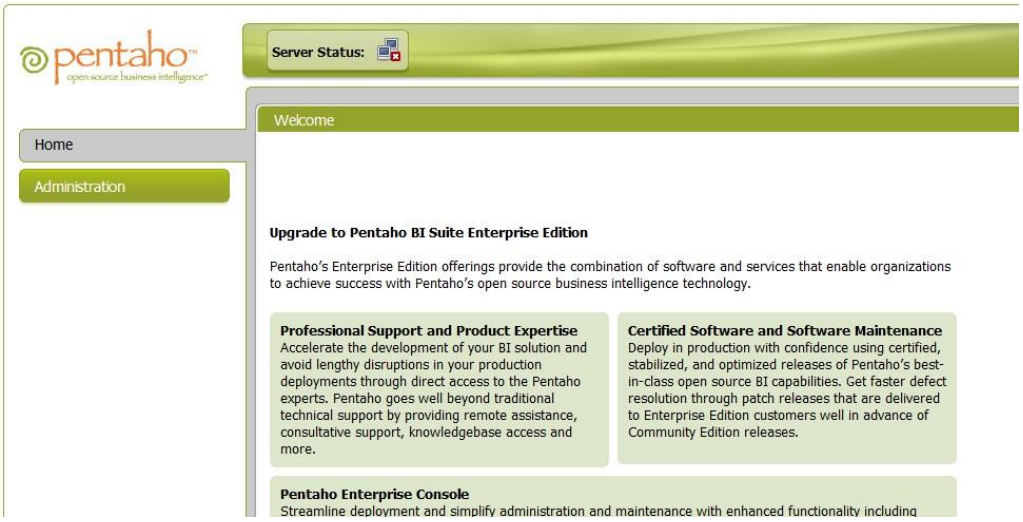
1. Download file di http://community.pentaho.com/projects/bi_platform/



2. Di dalam folder pentaho BI Server tersebut terdapat dua folder : administration-console, dan biserver-ce. Administrator console adalah bagian yang menangani hal-hal administrasi Pentaho (datasource, penjadwalan dan lainnya). Sedangkan biserver-ce adalah bagian yang menangani aplikasi BI itu sendiri sebagai platformnya.



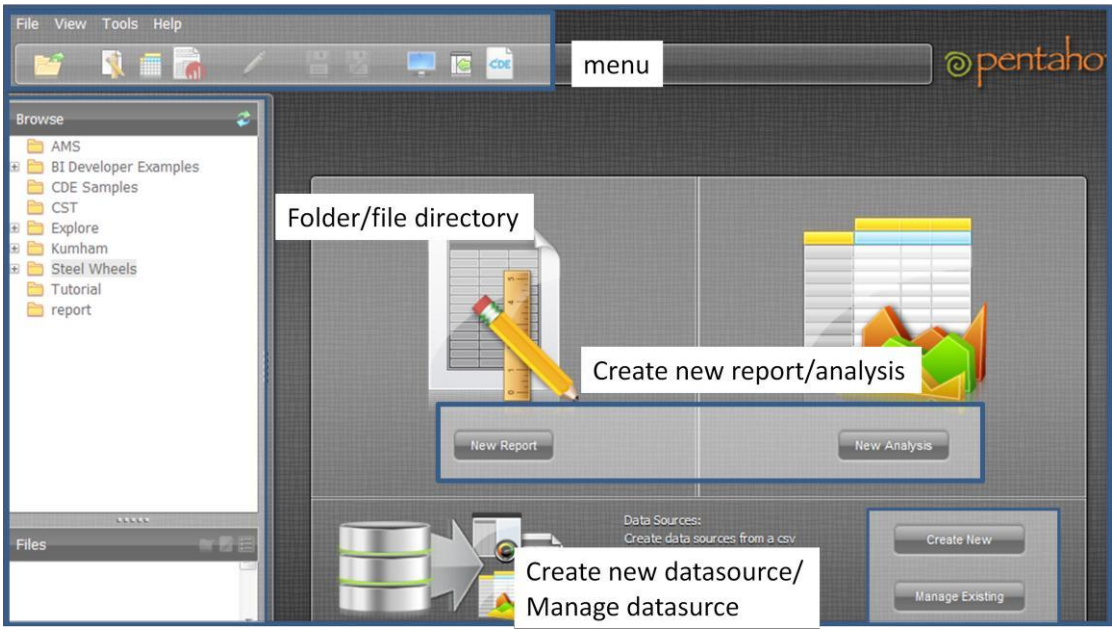
3. Untuk menjalankan administration-console, masuk ke folder adminsitration- console lalu jalankan start-pac.bat (untuk OS Windows) atau jalankan start- pac.sh (untuk OS Linux). Sebelumnya jalankan set-pentaho-env.bat atau set- pentaho-env.sh untuk konfigurasi environment secara otomatis. Lalu buka browser dengan alamat http://localhost:8099, dengan username : admin dan password : password. Maka muncul tampilan dari administration-console.



4. Untuk menjalankan BI Server, masuk ke folder biserver-ce lalu jalankan start- pentaho.bat atau start-pentaho.sh dengan sebelumnya menjalankan set- pentaho-env.bat atau set-pentaho-env.sh. Setelah itu, buka

browser dengan alamat http://localhost:8080. Maka akan muncul halaman dari Pentaho User Console.

3.2 Bagian-bagian dari Pentaho User Console



3.2.1 Menu

Di dalam bagian menu terdapat menu lists dan menu icon. Menu list mulai dari file hingga help. Dengan menu yang ada, kita dapat membuat, membuka datasource, dashboard, report, analysis view, dan juga plugin yang terinstal di Pentaho.

3.2.2 Folder/File Directory

Di bagian ini, kita bisa mengelola file dan folder tempat kita menyimpan dashboard, report, analysis view dan lainnya.

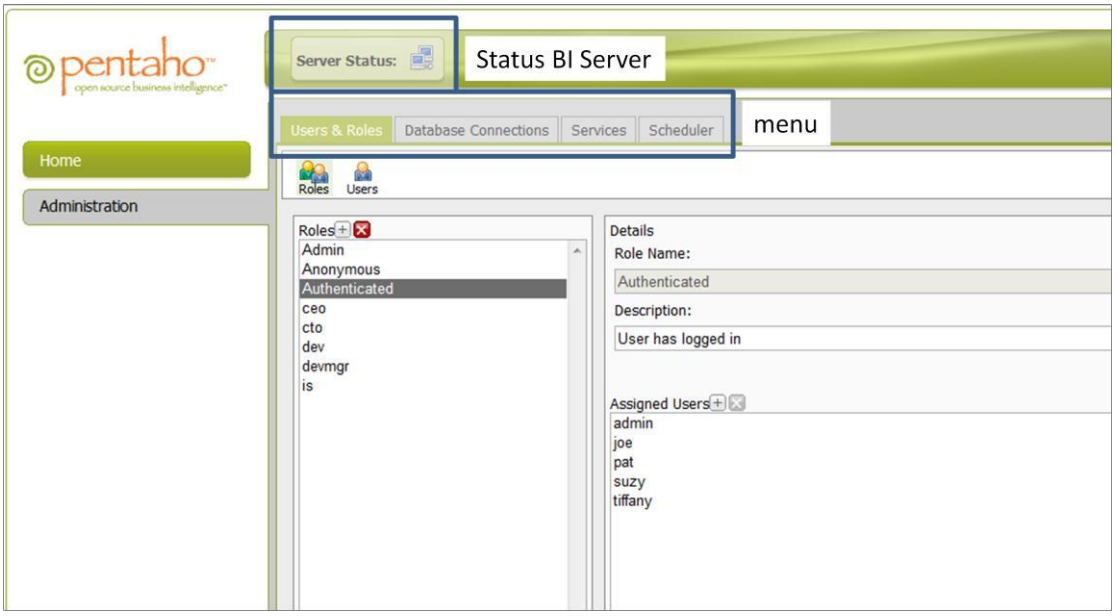
3.2.3 Create new report/analysis

Dengan meng-klik New Report atau New Analysis, akan muncul tampilan dari halaman report designer ataupun analysis view. Saat ini, Pentaho tidak lagi melakukan pengembangan maupun support kepada product ad hoc report dan analysis view.

3.2.4 Create new datasource/manage datasource

Salah satu kelebihan dari pentaho adalah memfasilitasi koneksi ke banyak jenis DBMS sehingga tidak ada kendala tentang jenis datasource yang digunakan. Create new datasource adalah fasilitas untuk mendefinisikan datasouce sedangkan manage datasource adalah fasilitas untuk mengelola datasource yang telah didefinisikan.

3.3 Bagian-bagian dari Administration Console



3.3.1 **Status BI Server**

Icon ini menampilkan apakah BI Server dalam keadaannya hidup atau tidak.

3.3.2 **Menu**

Di bagian ini, kita dapat mengelola *user* dan *role*-nya, *database connection*, *service* dan *scheduler*. Menu yang paling sering dikonfigurasi data *users* dan *database connection.*

Bab 4 Pentaho Data Integration/KETTLE

Seperti penjelasannya sebelumnya, salah satu produk unggulan dari Pentaho versi community adalah Pentaho Data Integration/KETTLE. Kettle dapat melakukan extract dan load dari banyak jenis DBMS. Kettle juga mempunyai 'Steps' yang cukup lengkap untuk membuat sebuah TRANSFORMATION ataupun JOB.

4.1 Definisi

Di dalam pembahasan Pentaho Data Integration akan muncul beberapa istilah, antara lain :

- Extract : Proses pengambilan data dari datasource
- Transform :Proses pengubahan data yang telah di*extract*
- Load : Proses store/penyimpanan data yang telah di*transform.*
- Job :file yang berekstensi .kjb yang berfungsi sebagai workflow control

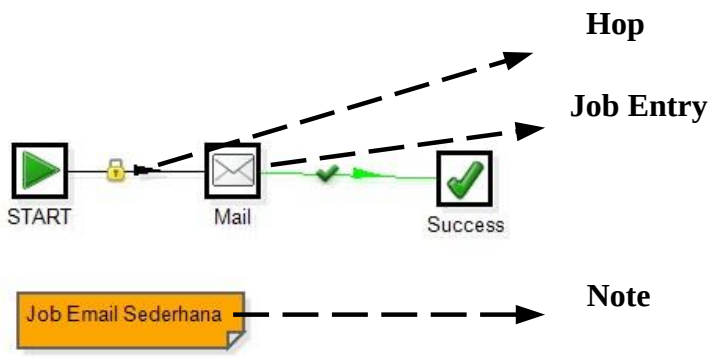
- Transformation : file yang berekstensi .ktr yang berfungsi sebagai data workflow.
- Kettle : Nama lain dari Pentaho Data Integration
- Spoon : Aplikasi GUI untuk merancang atau menjalankan job/transformation
- Pan : Utilitas untuk menjalanan transformation dalam tampilan console. Biasanya digunakan untuk otomasi terjadwal.
- Kitchen : Utilitas untuk menjalankan job dalam tampilan console. Biasanya digunakan untuk otomasi terjadwal.
- Step : tahapan yang ada pada perancangan transformation/job
- Row : baris dari data yang berisi 0 data atau lebih dari value
- Value : bagian dari row dan dapat berisi tipe data apa pun (string, float, integer, dan lainnya)
- Output stream : rangkaian row (baris) yang meninggalkan sebuah step
- Input stream : rangkaian row (baris) yang memasuki sebuab step
- Hop : garis yang merepresentasikan satu atau lebih dari data stream di antara dua step
- Note : deskripsi yang diberikan/didefinisikan terhadap sebuah proses pada transformation/job
- Job Entry: A job entry is one part of a job and performs a certain task

4.4 Simulasi Penggunaan Kettle

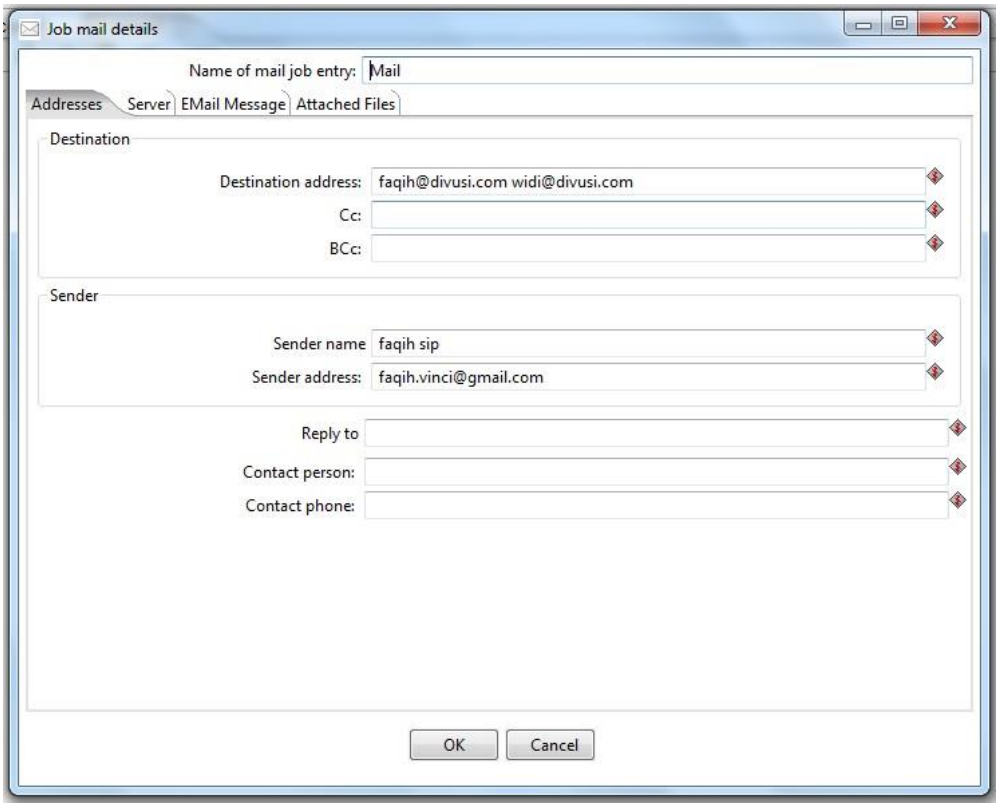
4.4.1 Simulasi Job

4.4.1.1 Job Email sederhana

Idenya adalah mengirimkan pesan email melalui Kettle ke alamat yang dituju. Berikut rangkaian step yang dibangun.



Tab Address -> berisi data email tujuan dan pengirim



Destination address : alamat tujuan

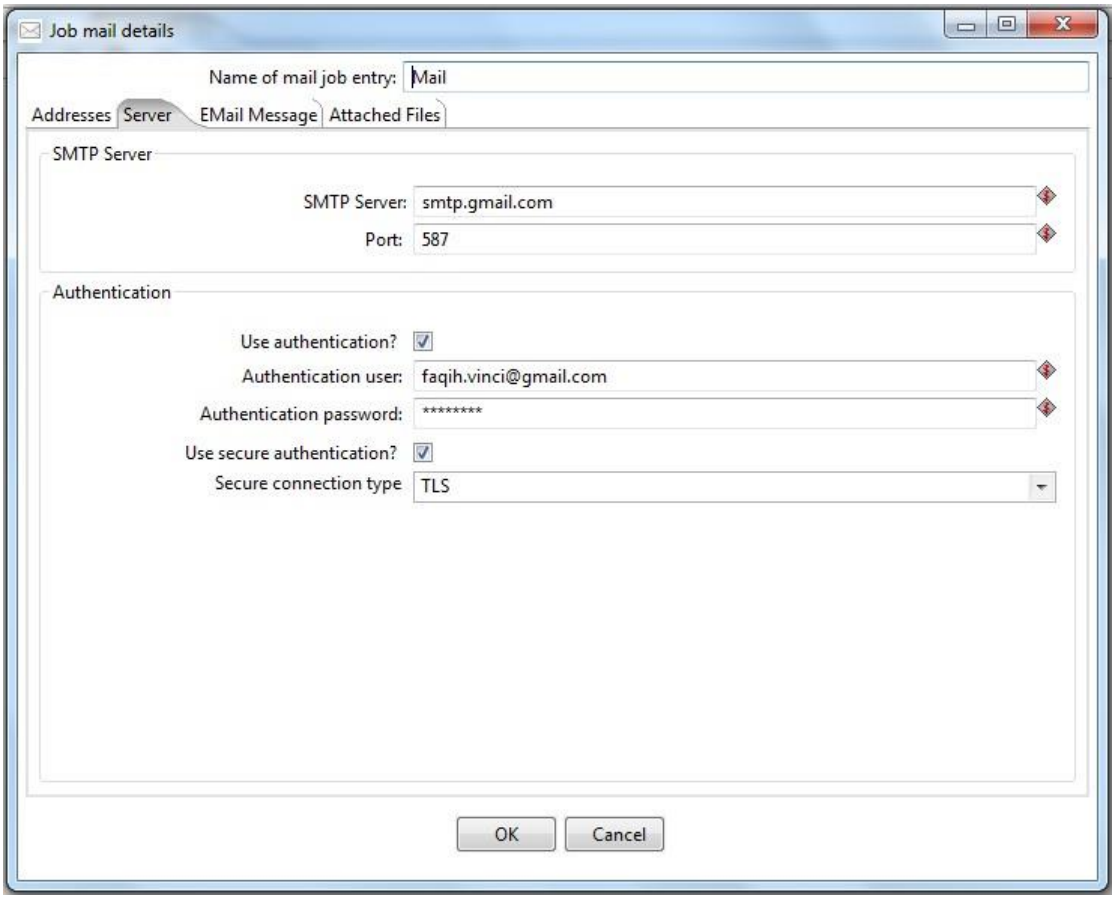
Cc dan BCc :

Sender name : berisi nama dari pengirim

Sender address : berisi alamat email dari pengirim

Reply to : alamat untuk membalas email

Tab Server -> Berisi data sestting server pengirim



SMTP server : berisi alamat smtp/email sender setting. Isinya tergantung dari email yang digunakan

Port : berisi port yang digunakan. Hal ini juga tergantung dari email yang digunakan

User authentication : jika dichecklist maka akan menggunakan otentifikasi

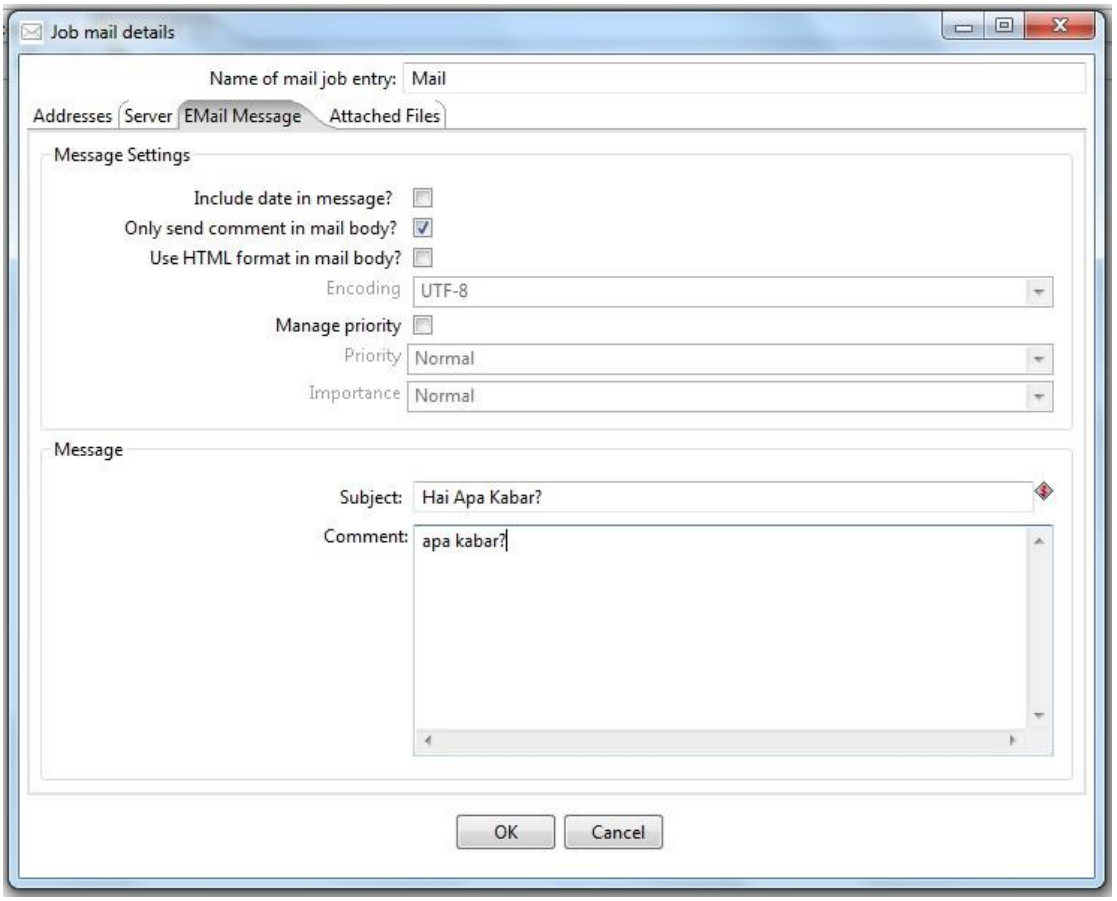
Authentication user : berisi email pengirim

Authentication password : berisi password email pengirim

Use secure authentication : menggunakan security otentifikasi

Secure connection type : tipe konesi sekuriti. Biasanya digunakan TLS (Transport Layer Security)

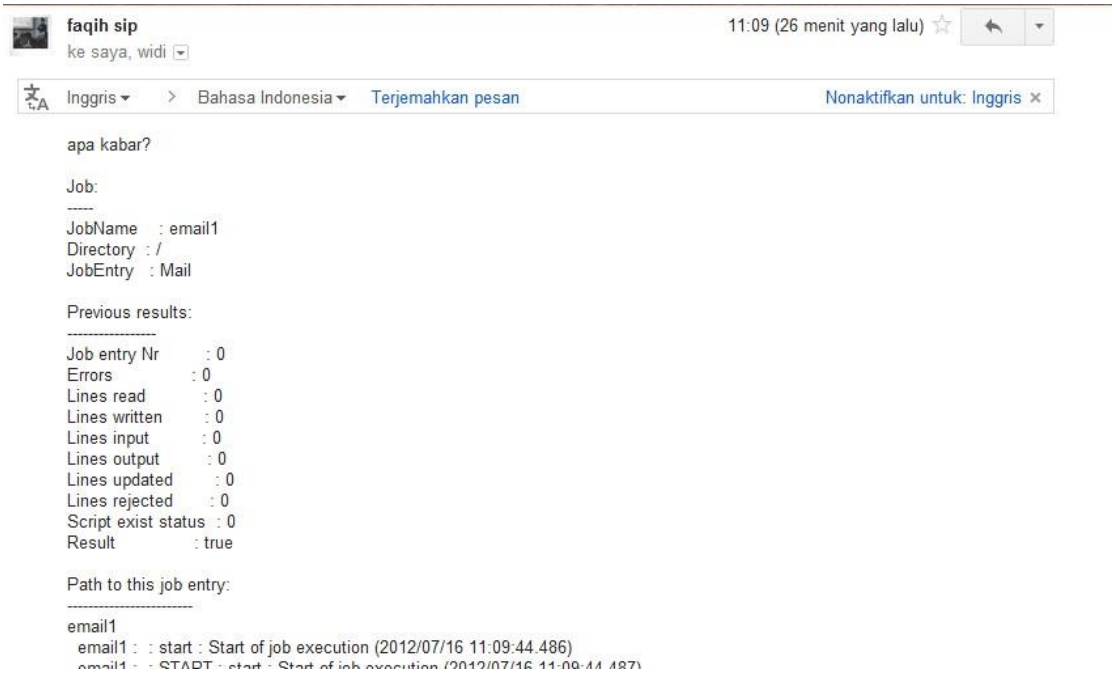
Tab Email Message -> berisi konten dari email yang akan dikirim



Message setting berfungsi untuk mensetting pesan yang akan dikirim.

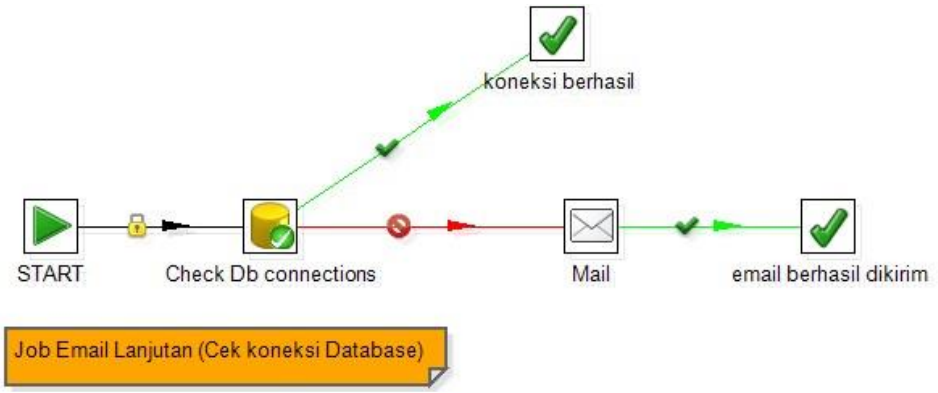
- Include date in message : memasukan tanggal proses dijalankannya Job ke dalam email
- Only send commect in mail body : untuk memasukkan hanya comment/isi dari email. Jika tidak dichecklist, maka di email yang dikirim akan muncul log dari proses ETL
- Use HTML format in mail body : untuk mengaktifkan pembacaan tag HTML
- Manage priority : untuk mengelola prioritas pengiriman email
- Subject : subjek dari email
- Comment : konten dari email

Jika kita jalankan email tersebut, maka pada email penerima akan menerima email yang berisi konten yang kita inginkan.

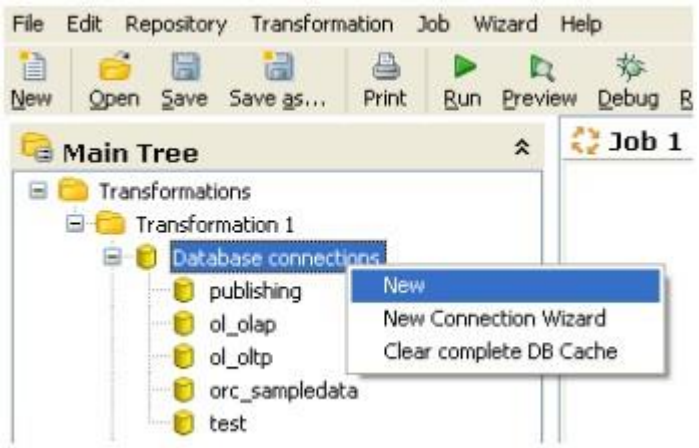


4.4.1.2 Job Email Lanjutan

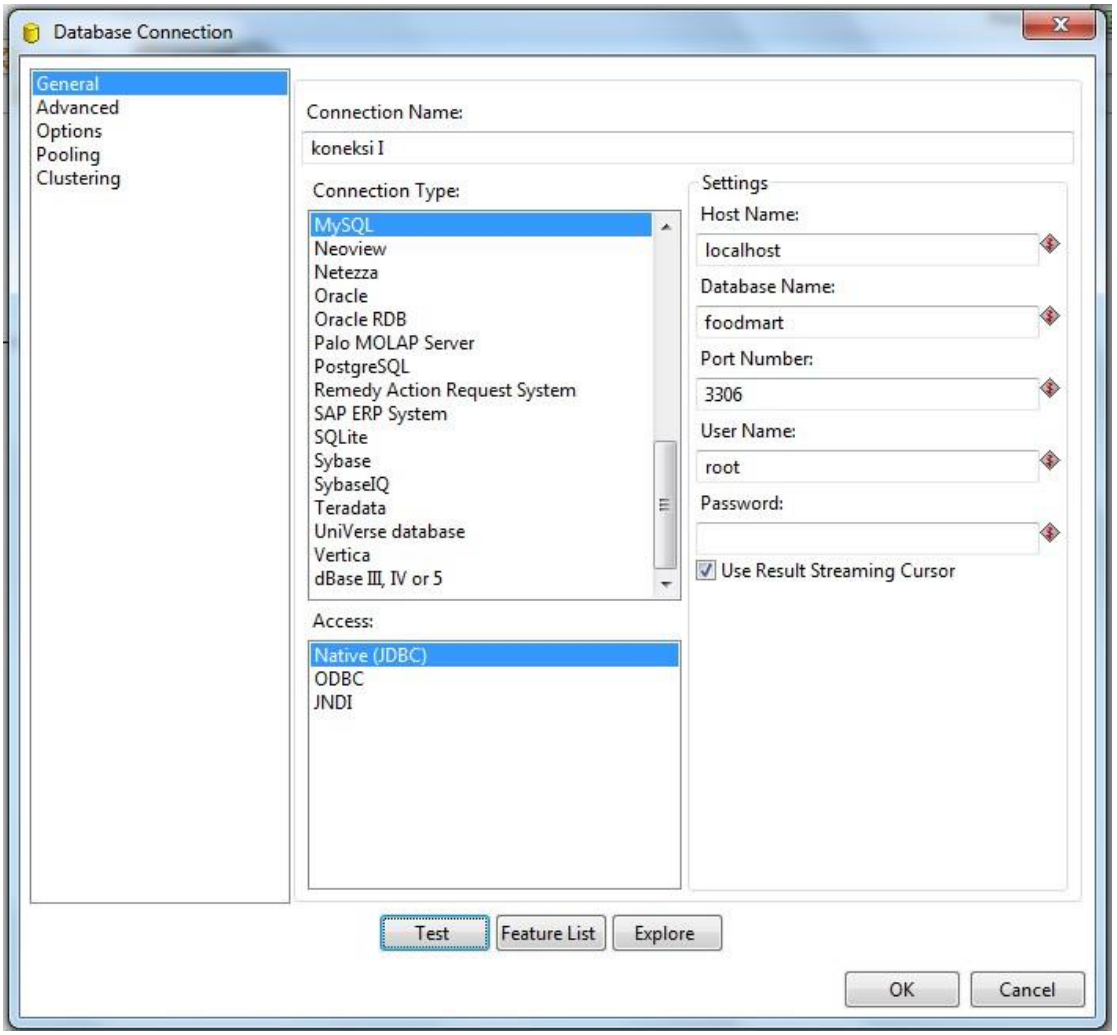
Kali ini akan dicoba membuat Job yang berfungsi sebagai pemberitahuan bahwa koneksi ke database tertentu mengalami gangguan.



Sebelumnya dibuat sebuah koneksi ke database tertentu.



Caranya adalah pada panel kiri buka tab view, pada List Database connection klik kanan dan pilih new.

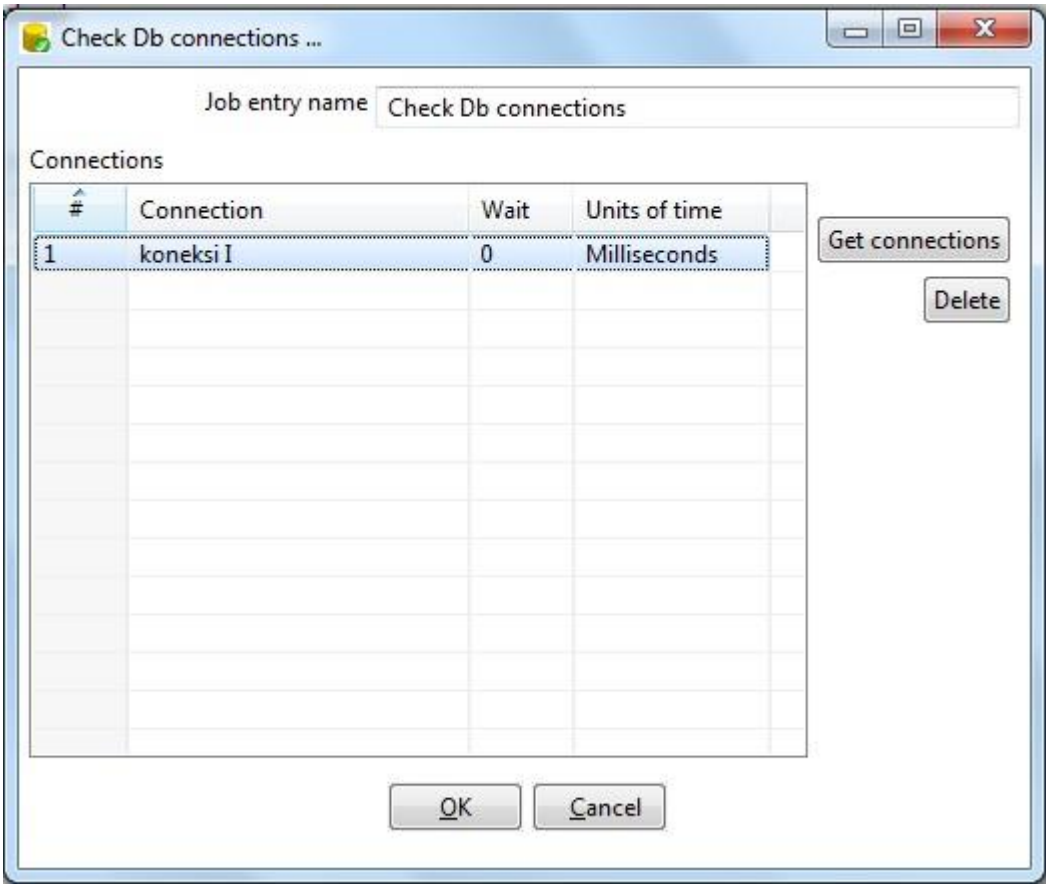


Isikan semua field yang ada, lalu klik test untuk menguji koneksi. Jika berhasil maka akan muncul notifikasi berhasil.



Klik OK pada kotak dialog yang ada. Sekarang kita suda mempunyai sebuah koneksi ke localhost dengan database foodmart.

Kembali ke pembahasan sebelumnya. Kita double click step *Check Db Connections.*



Isikan field connection dengan Koneksi I, lalu klik OK.

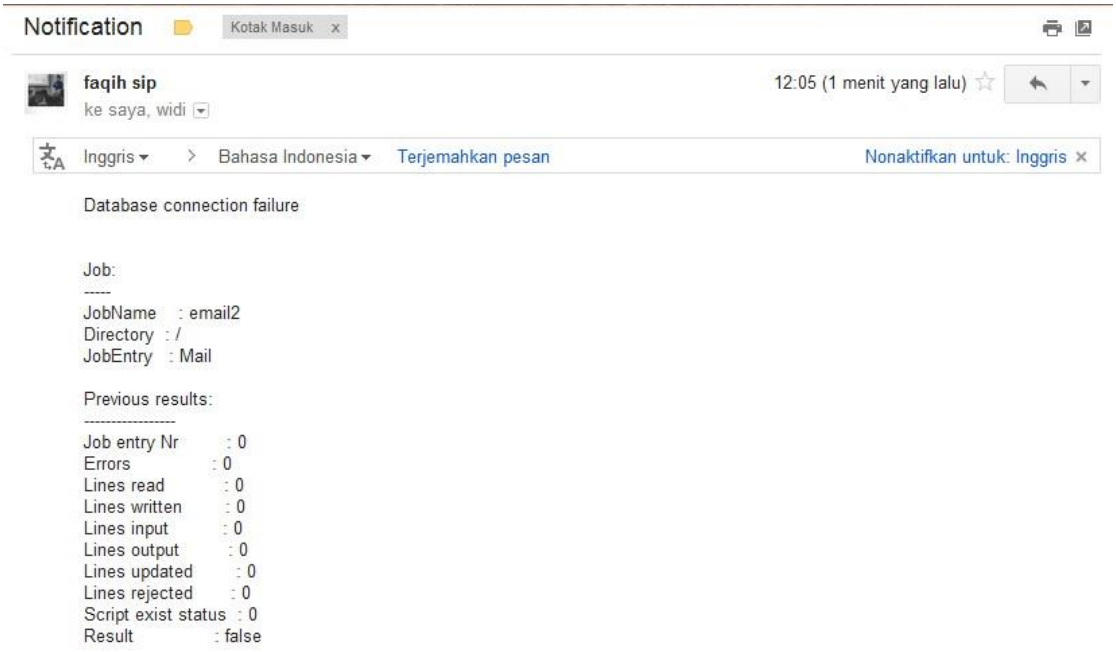
Sekarang kita akan menjalankan Job email2 ini. Dengan kondisi yang ada, maka seharusnya akan masuk ke Step Koneksi berhasil seperti di bawah ini.



Kita akan mencoba scenario gagal dengan mengubah nama database menjadi foodmart1. Maka akan masuk ke Step pengiriman email.

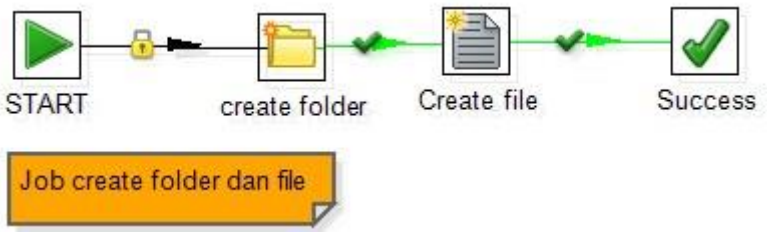


Jika kita buka email penerima notifikasi, maka akan ada email seperti berikut.



4.4.1.3 Job Create Folder dan File

Job dapat juga melakukan pembuatan folder dan file. Berikut rancangan dari J o b .



4.4.2 Simulasi Transformation

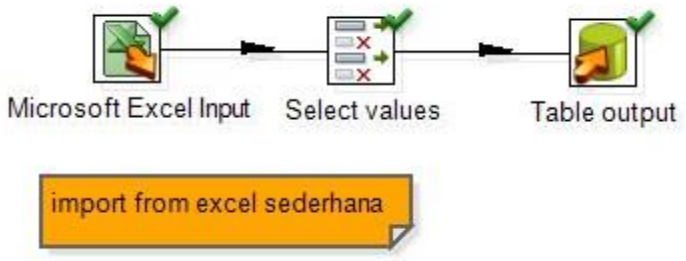
4.4.2.1 Transformation Extract dari Excell ke Database (kasus sederhana)

Pentaho Data integration dapat mengolah data yang berasal dari file bertipe excel yang kemudian disimpan di database. Kali ini kita akan bahas kasus sederhana mengimport dara dari file excel menuju database. Berikut contoh dari tabel pada excel yang akan kita extract lalu load ke databse :

No	Nama Daerah	Tebu	Kakao	Karet	Perikanan Tangkap	Kelapa Sawit	Tahun
1	Aceh	0	30339	84636	0	493826	2010
2	Bali	0	6177	85	104926	0	2010
3	Bangka0Belitung	0	981	18806	0	0	2010
4	Banten	0	2213	14861	0	25249	2010
5	Bengkulu	0	5297	48688	0	615624	2010

6	Daerah Istimewa Yogyakarta	16573	1243	0	0	0	2010
7	DKI Jakarta	0	0	0	0	0	2010
8	Gorontalo	4114	3799	0	0	0	2010
9	Jambi	0	532	290439	0	1293173	2010
10	Jawa Barat	98942	4176	52681	0	0	2010
11	Jawa Tengah	0	0	28163	0	0	2010
12	Jawa Timur	1109855	23166	26621	0	0	2010
13	Kalimantan Barat	0	3332	252604	0	881768	2010
14	Kalimantan Selatan	0	35	103563	0	434135	2010
15	Kalimantan Tengah	0	287	258641	0	2214093	2010
16	Kalimantan Timur	0	12552	24403	0	566687	2010
17	Kepulauan Riau	0	0	0	0	191	2010
18	Lampung	1017561	0	64188	0	373001	2010
19	Maluku	0	9688	0	0	0	2010
20	Maluku Utara	0	13689	0	0	0	2010
21	Nusatenggara Barat	0	0	0	0	0	2010
22	Nusatenggara Timur	0	12569	0	0	0	2010
23	Papua	0	11522	1558	0	34283	2010
24	Papua Barat	0	3047	12	0	64684	2010
25	Riau	0	0	0	0	6064391	2010
26	Sulawesi Barat	0	0	0	0	321671	2010
27	Sulawesi Selatan	30551	173555	5136	0	2881	2010
28	Sulawesi Tengah	0	144049	0	0	158202	2010
29	Sulawesi Tenggara	0	137833	0	0	0	2010
30	Sulawesi Utara	0	0	0	0	0	2010
31	Sumatera Barat	0	34806	88949	0	852042	2010
32	Sumatera Selatan	0	1781	515965	0	2082196	2010
33	Sumatera Utara	0	69978	413597	0	3230488	2010

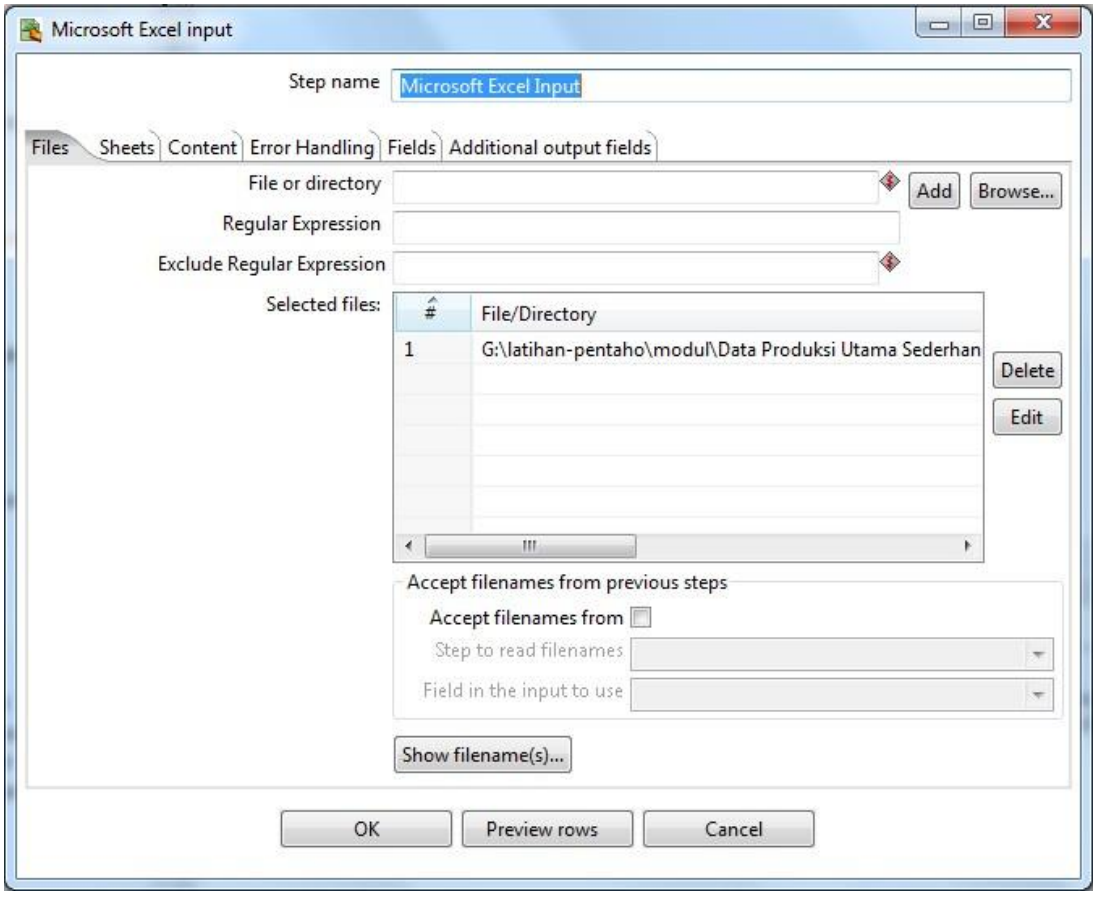
Untuk permasalahan ini, berikut disain dari transformation yang kita akan jalankan.



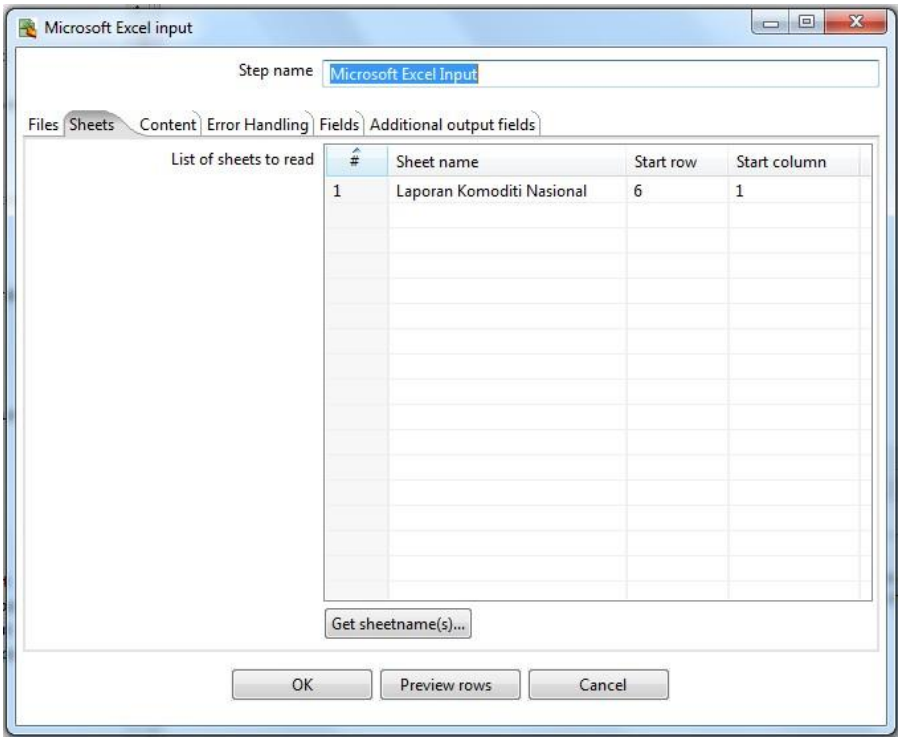
Keterangan

1. Step Microsoft Excel Input.

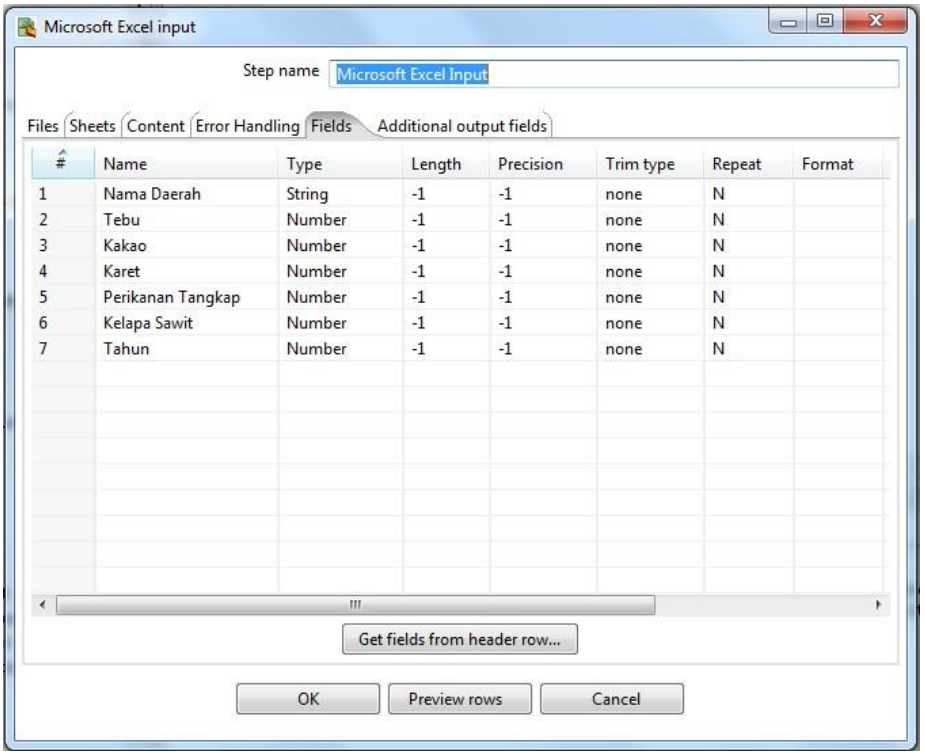
    a. Masukan alamat dari file yang akan kita exctact. (add Browse, pilih file, klik Add)



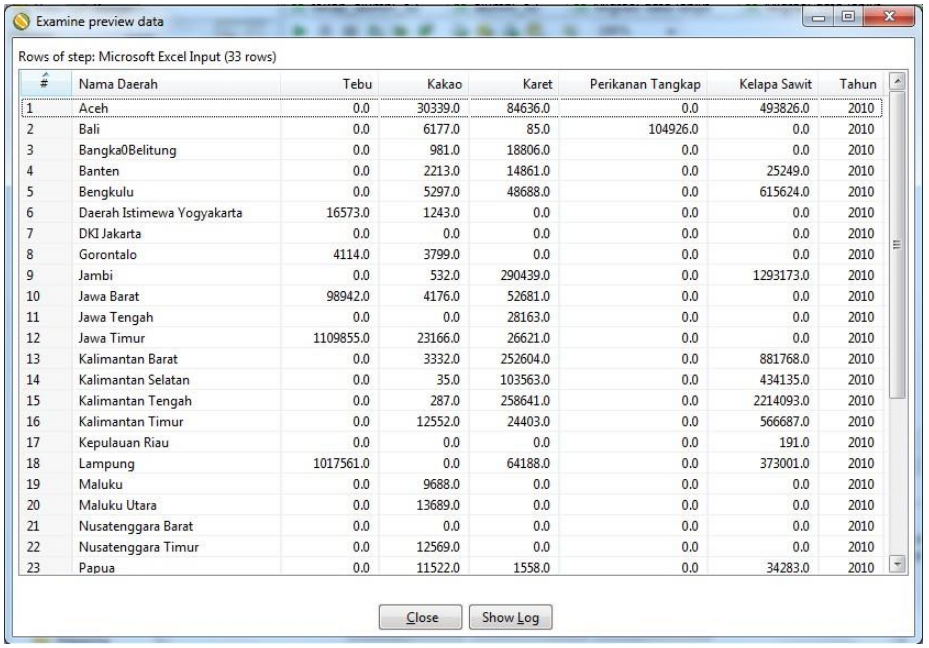
b. Pilih sheet yang akan kita gunakan dengan mengklik tombol Get Sheetname(s) lalu tentukan Start row dan start column.



c. Tentukan field yang akan diambil dengan mengklik tombol ‘get fields from header row’.

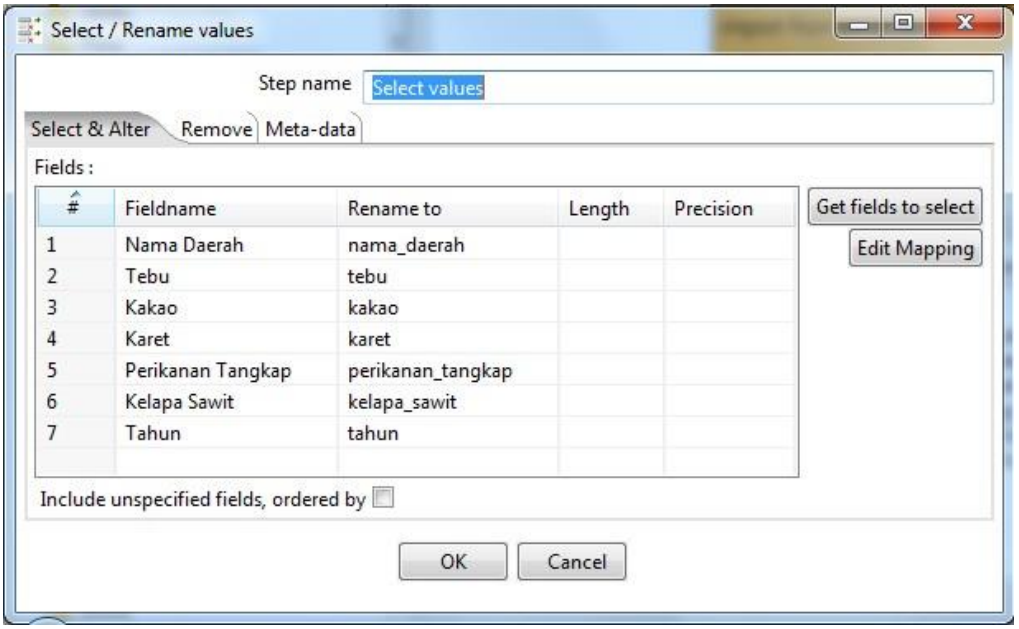


d. Setelah itu klik tombol 'Get fields from header row..'. maka akan muncul data yang akan kita ambil berdasarkan field yang kita tentukan.

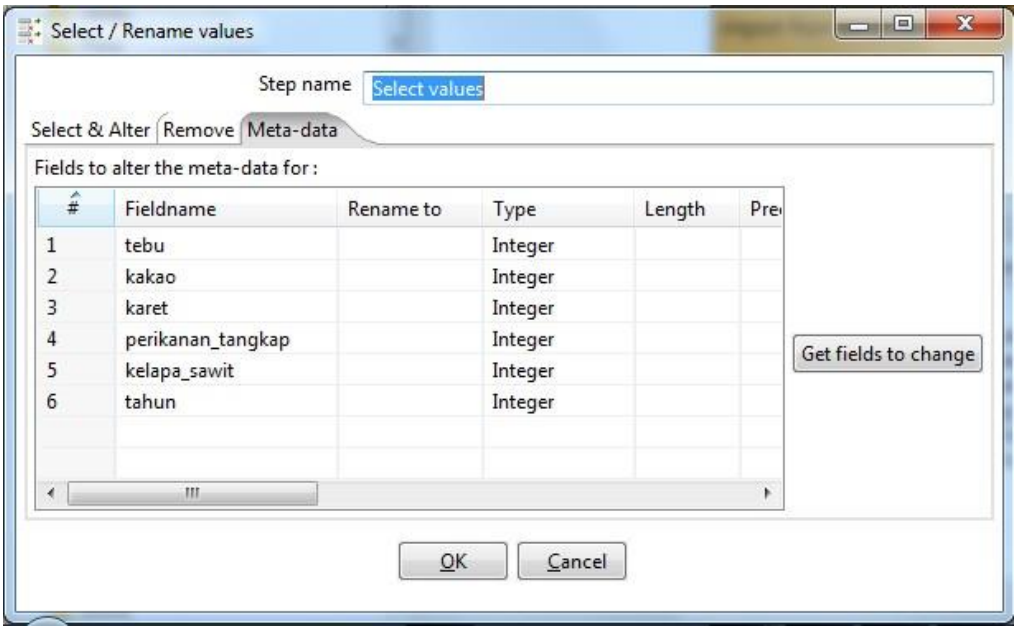


2. Pada step 'Select Value',

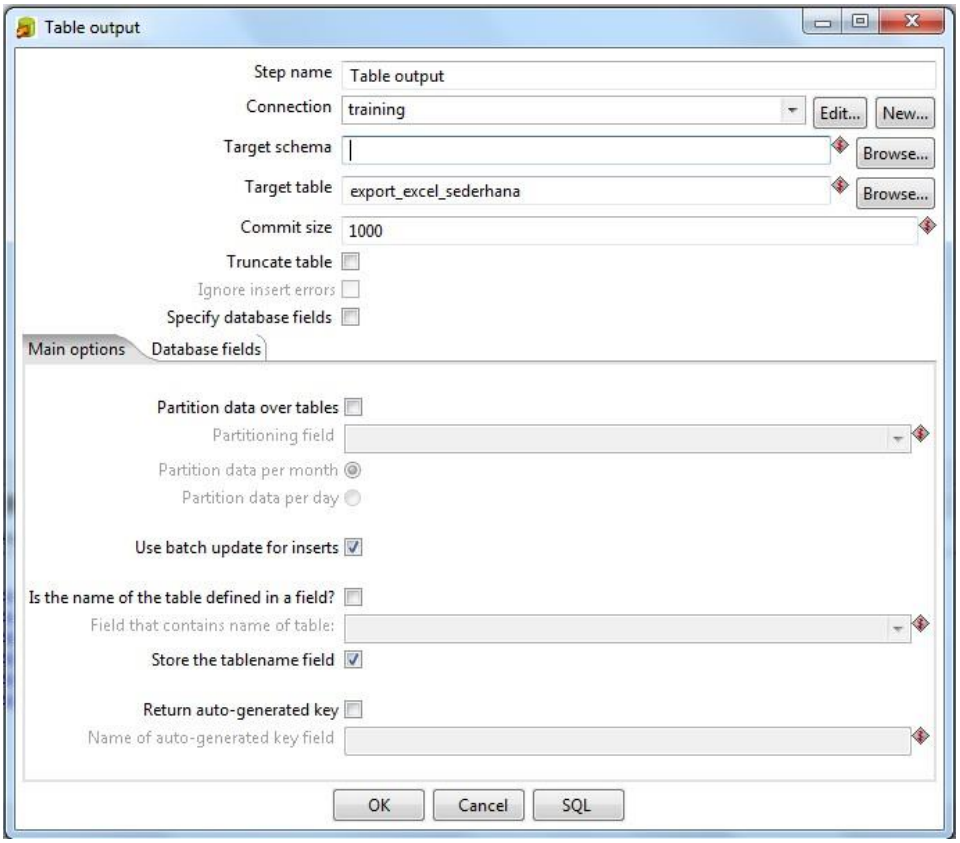
a. ubah nama field menjadi nama yang database friendly dengan mengubah spasi dengan garis bawah/underscore (_)



b. ubah metadata sesuai kebutuhan.



3. pada step ‘table output’, tentukan koneksi dan juga table destinasi. Jika tabel belum pernah dibuat, maka tekan tombol ‘SQL’ untuk meng-*generate* tabel tersebut



Lalu tekan tombol run atau F5 untuk menjalankan transformasi tersebut.

4.4.2.2 Transformation Extract dari Excell ke Database (kasus lanjut)

Kasus kali ini menggunakan data yang mirip dengan data sebelumnya. Namun ada permasalahan bentuk datanya. Angka per seribu didahului oleh titik, dan data yang 0 ditulis '-'. Jika kita perlakukan sama dengan kasus sebelumnya maka akan terjadi error.

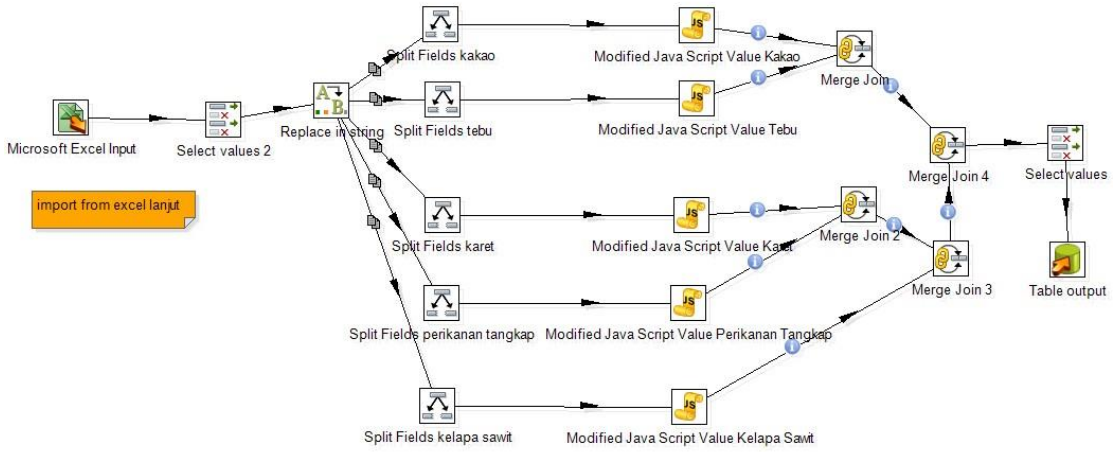
Solusinya adalah kita tangani tanda titik agar menghilang, dan mengubah '-' menjadi 0. Menghilangkan tanda titik menggunakan Step 'Split Fields' menyebabkan pertambahan jumlah filed. Masing-masing field tersebut harus digabung kembali agar menjadi satu field yang sudah tidak ada tada titik.

Contoh data yang harus diolah lebih lanjut :

No	Nama Daerah	Tebu	Kakao	Karet	Perikanan Tangkap	Kelapa Sawit	Tahun
1	Aceh	-	30.339	84.636	-	493.826	2010
2	Bali	-	6.177	85	104.926	-	2010
3	Bangka-Belitung	-	981	18.806	-	-	2010
4	Banten	-	2.213	14.861	-	25.249	2010
5	Bengkulu	-	5.297	48.688	-	615.624	2010
6	Daerah Istimewa Yogyakarta	16.573	1.243	-	-	-	2010
7	DKI Jakarta	-	-	-	-	-	2010
8	Gorontalo	41.14	3.799	-	-	-	2010
9	Jambi	-	532	290.439	-	1.293.173	2010
10	Jawa Barat	98.942	4.176	52.681	-	-	2010
11	Jawa Tengah	-	-	28.163	-	-	2010
12	Jawa Timur	1.109.855	23.166	26.621	-	-	2010
13	Kalimantan	-	3.332	252.604	-	881.768	2010

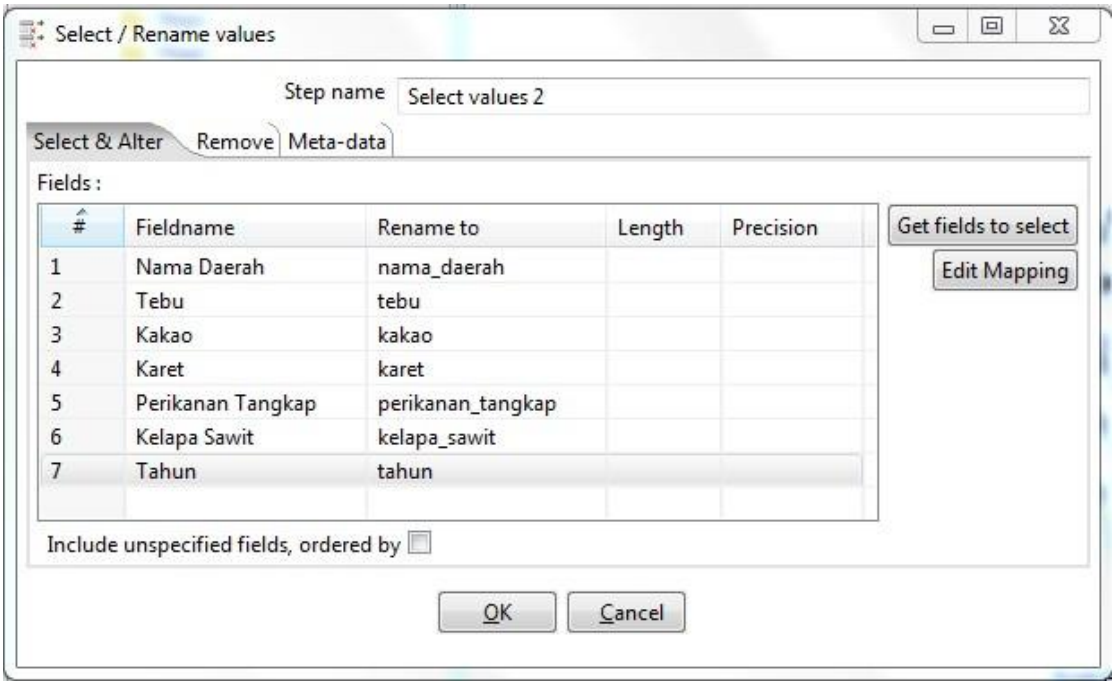
14	Kalimantan Selatan	-	35	103.563	-	434.135	2010
15	Kalimantan Tengah	-	287	258.641	-	2.214.093	2010
16	Kalimantan Timur	-	12.552	24.403	-	566.687	2010
17	Kepulauan Riau	-	-	-	-	191	2010
18	Lampung	1.017.561	-	64.188	-	373.001	2010
19	Maluku	-	9.688	-	-	-	2010
20	Maluku Utara	-	13.689	-	-	-	2010
21	Nusatenggara Barat	-	-	-	-	-	2010
22	Nusatenggara Timur	-	12.569	-	-	-	2010
23	Papua	-	11.522	1.558	-	34.283	2010
24	Papua Barat	-	3.047	12	-	64.684	2010
25	Riau	-	-	-	-	6.064.391	2010
26	Sulawesi Barat	-	-	-	-	321.671	2010
27	Sulawesi Selatan	30.551	173.555	5.136	-	28.81	2010
28	Sulawesi Tengah	-	144.049	-	-	158.202	2010
29	Sulawesi Tenggara	-	137.833	-	-	-	2010
30	Sulawesi Utara	-	-	-	-	-	2010
31	Sumatera Barat	-	34.806	88.949	-	852.042	2010
32	Sumatera Selatan	-	1.781	515.965	-	2.082.196	2010
33	Sumatera Utara	-	69.978	413.597	-	3.230.488	2010

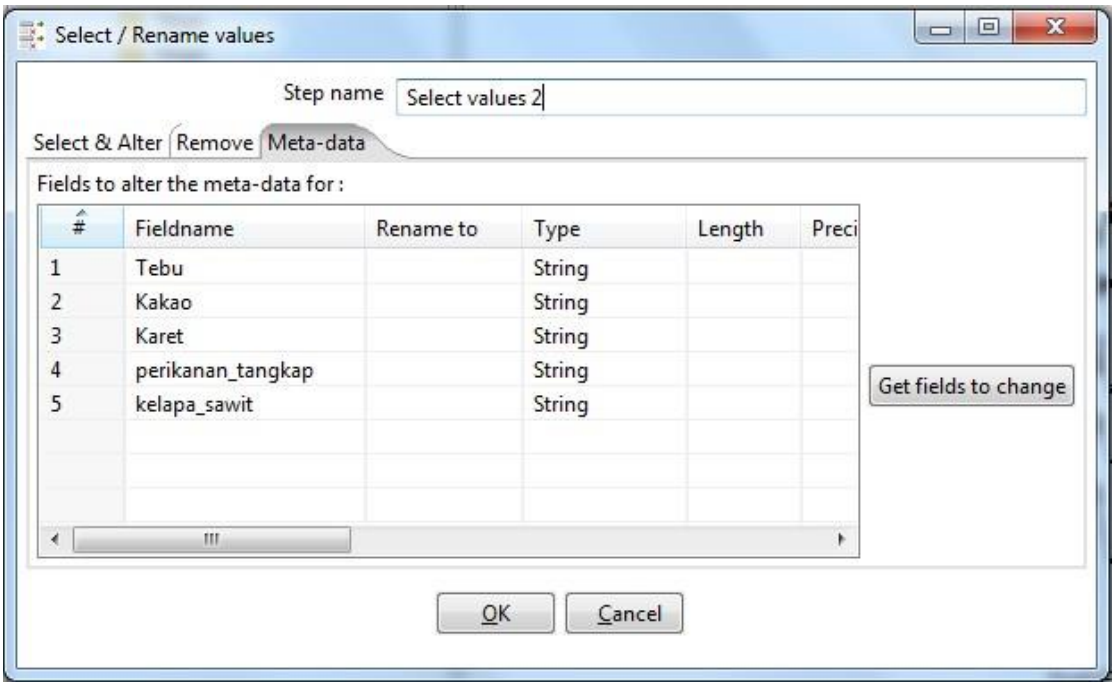
Untuk permasalahan ini, berikut disain dari transformation yang kita akan jalankan.



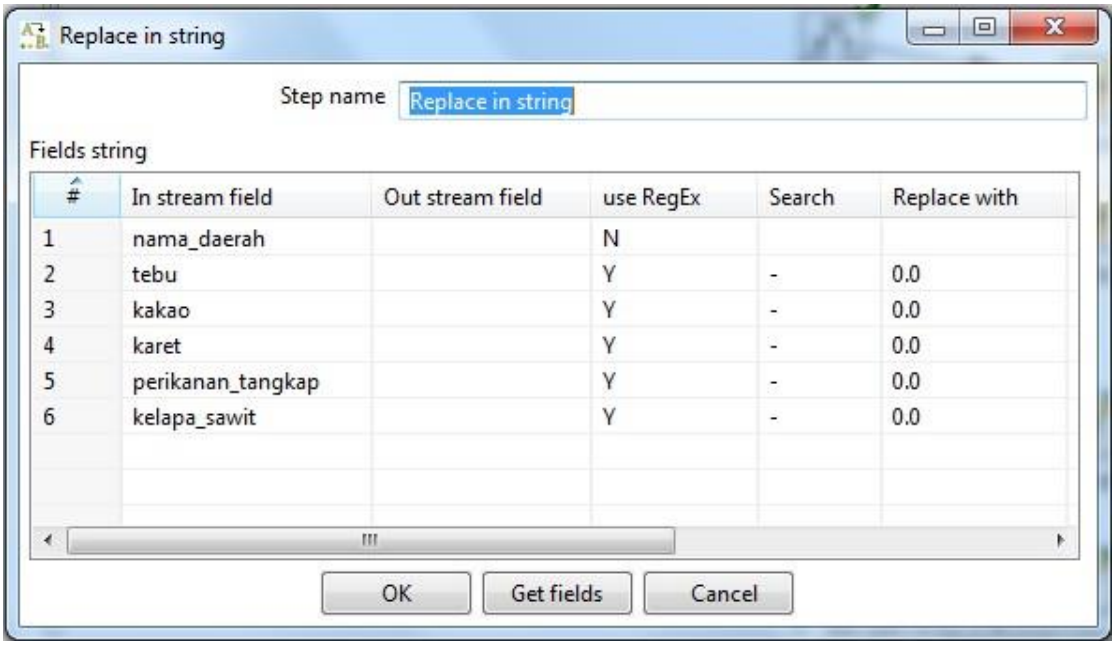
1. Langkah pertama, sama dengan contoh yang sederhana.

2. Langkah kedua, ubah metadata menjadi string agar dapat diolah secara string pada Step berikutnya dan mengubah nama field menjadi nama yang database friendly, yaitu mengubah spasi menjadi tanda '_' (underscore).

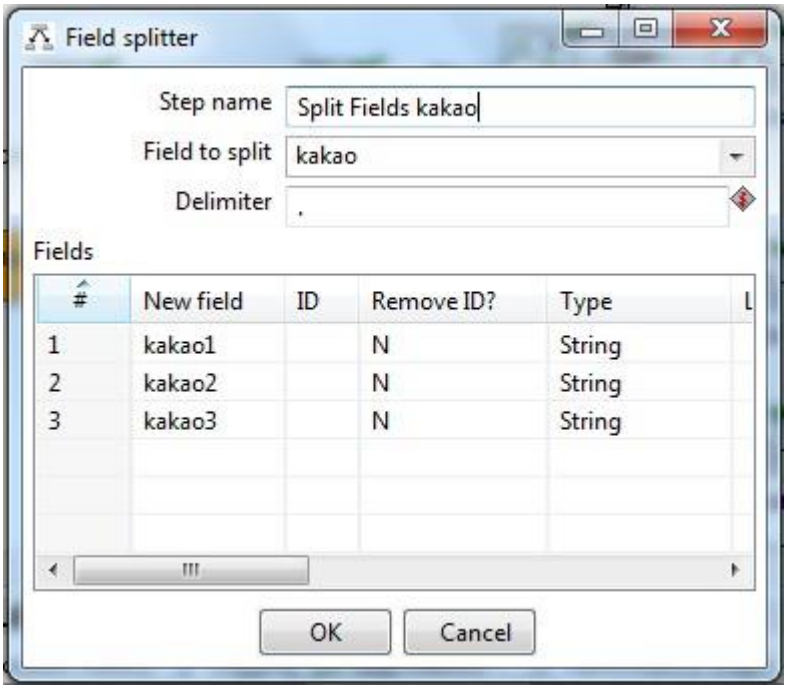




3. Langkah ketiga, step ‘Replace Sting’ berfungsi untuk mengubah tanda ‘-’ menjadi 0.



4. Berikutnya, lakukan pemecahan nilai dengan pemisah (delimiter) '.' dengan tujungan menghilangkan tanda titik '.' . Hasil dari step 'Replace in string' ini adalah field baru hasil pemecahan tersebut.

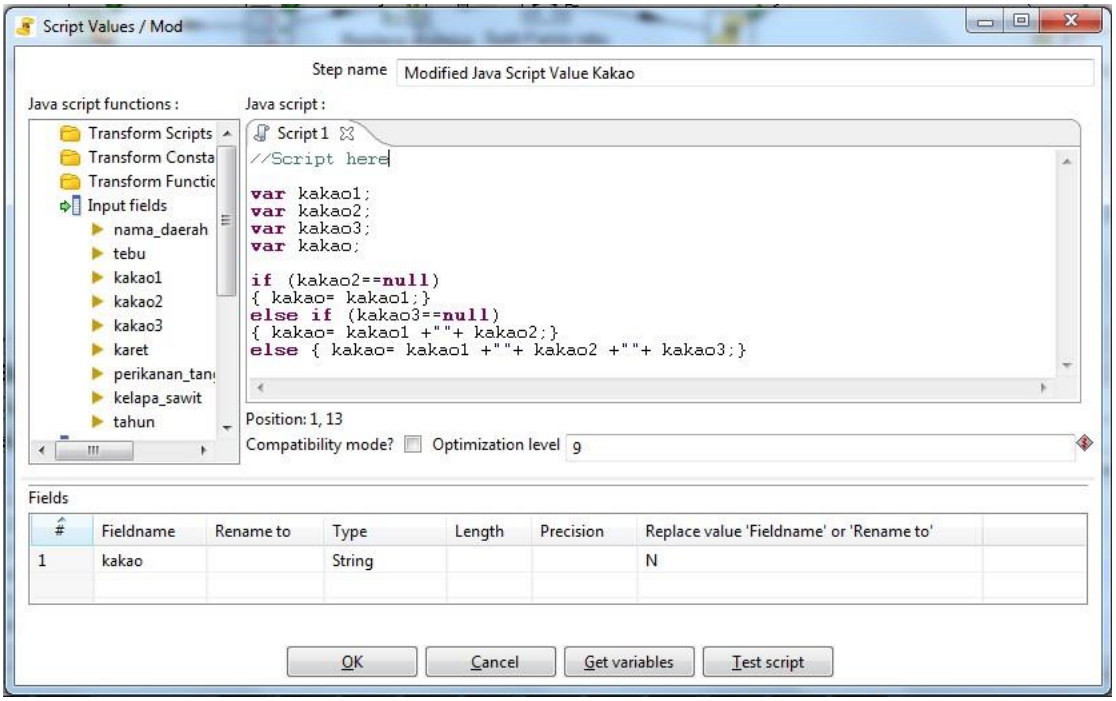


5. Kita lakukan penyatuan field-field dari hasil step sebelumnya dengan menggunakan step 'Script Value/Mod'. Hasil dari proses ini adalah field yang berisi field-field yang sebelumnya dipecah. Berikut adalah script yang dapat digunakan :

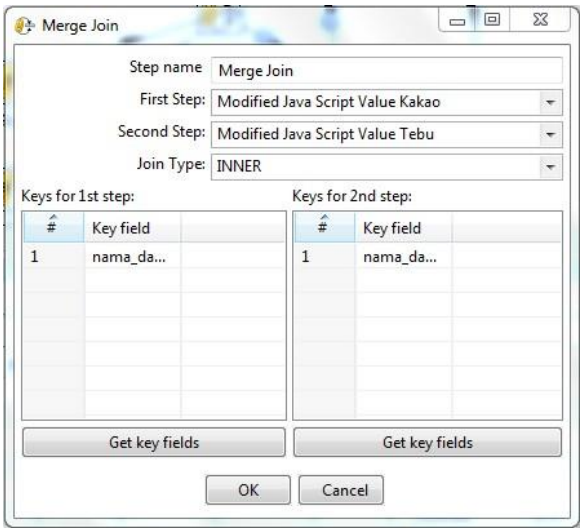
```
var kakao1;
var kakao2;
```

```
var  kakao3;
var kakao;

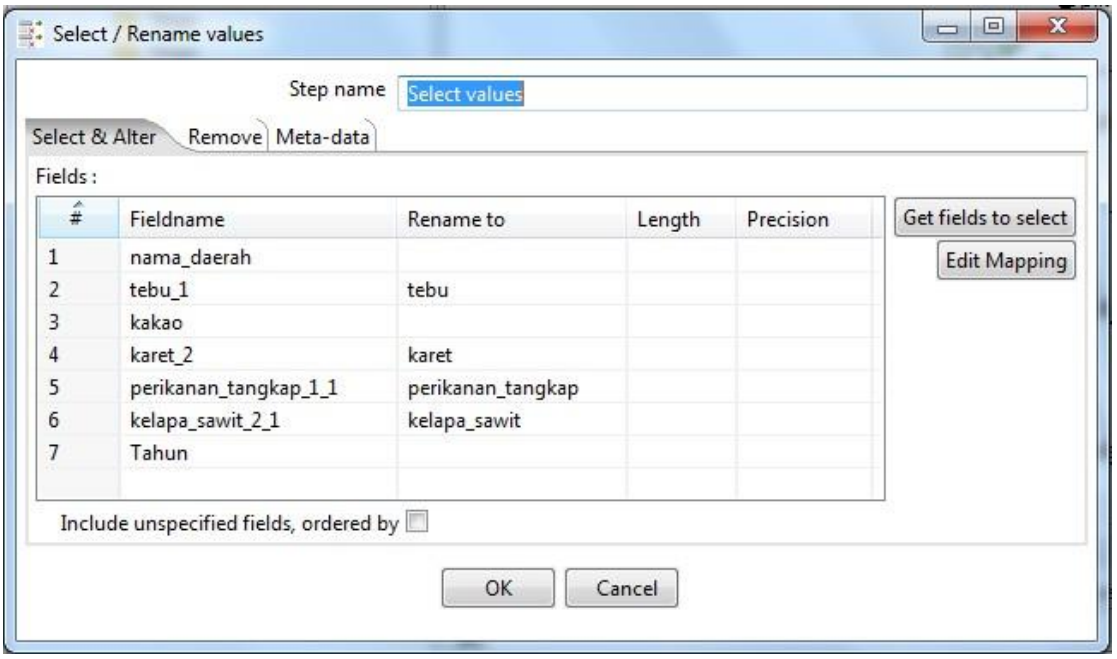
if (kakao2==null)
{ kakao= kakao1;}
else if (kakao3==null)
{ kakao= kakao1 +""+ kakao2;}
else { kakao= kakao1 +""+ kakao2 +""+ kakao3;}
```



6. Lakukan join hasil masing-masing step sebelumnya dengan step '**Merge join**'.



7. pilih value (Select Value) yang akan menjadi output dari transformasi ini, yaitu field yang bentuk datanya sudah dianggap benar. Setelah itu lakukan renaming agar namanya menjadi lebih baik.

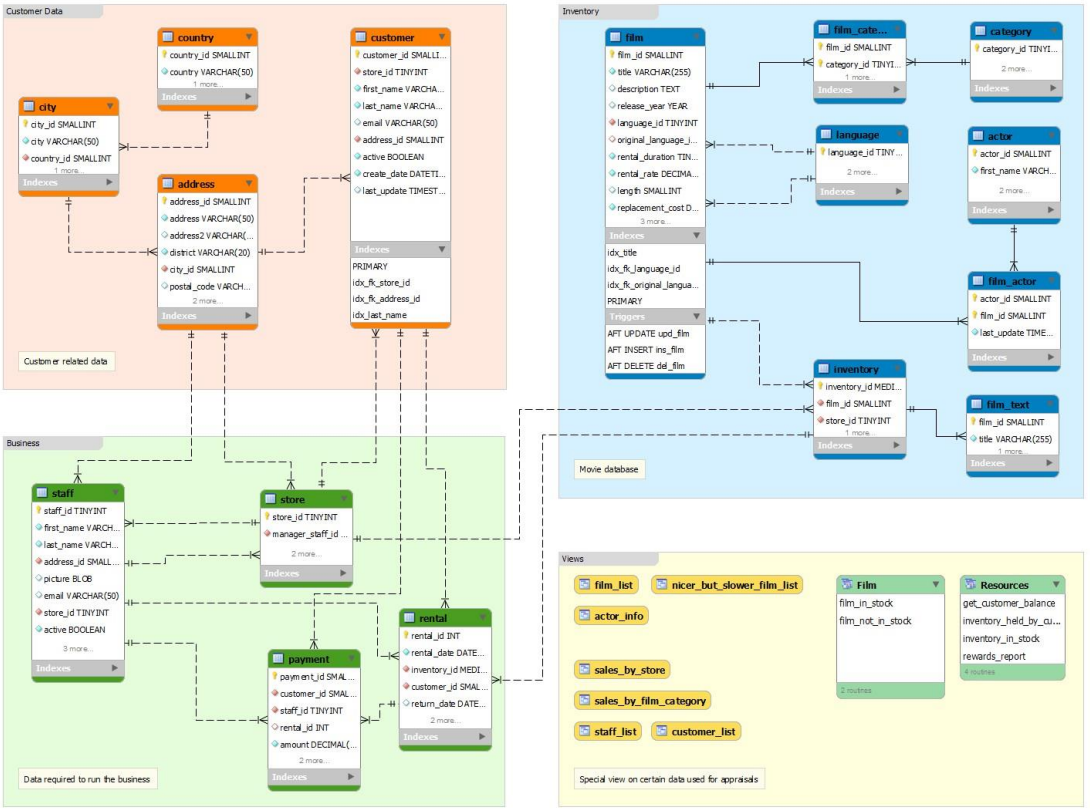


8. Terakhir gunakan tabel output untuk menyimpannya ke database yang diinginkan.

4.4.2.3 Transformation Extract dari Database ke Database

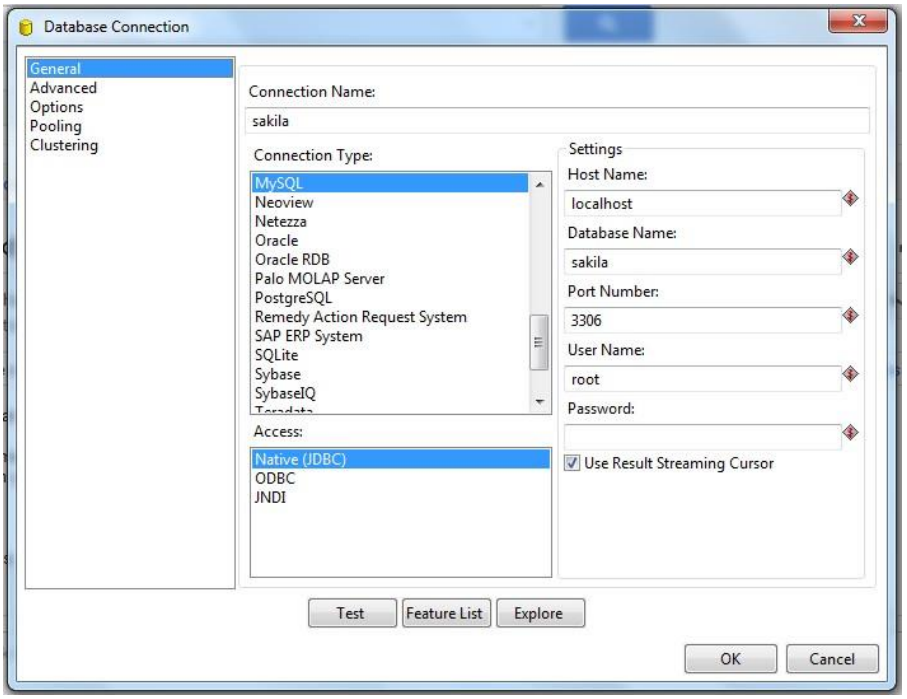
Salah satu keunggulan utama dari PDI adalah dapat mengekstrak dan meload dari/ke banyak jenis database. Kita akan mencoba mengekstrak data dari database Sakila ke database training. Sakila adalah salah satu database dummy tentang rental film.

Sebelumnya, buat dulu database sakila dengan menjalankan sql yang terlampir (skema dan datanya). Setelah itu, buat database tujuan dengan nama ‘training’.

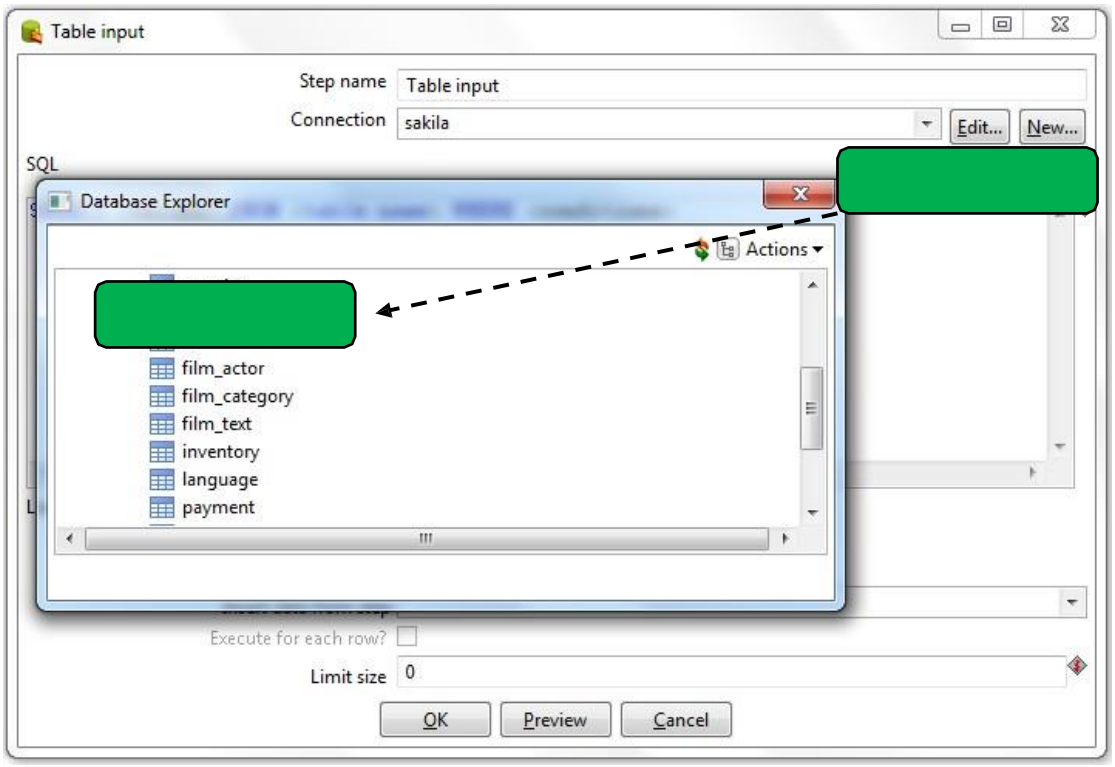




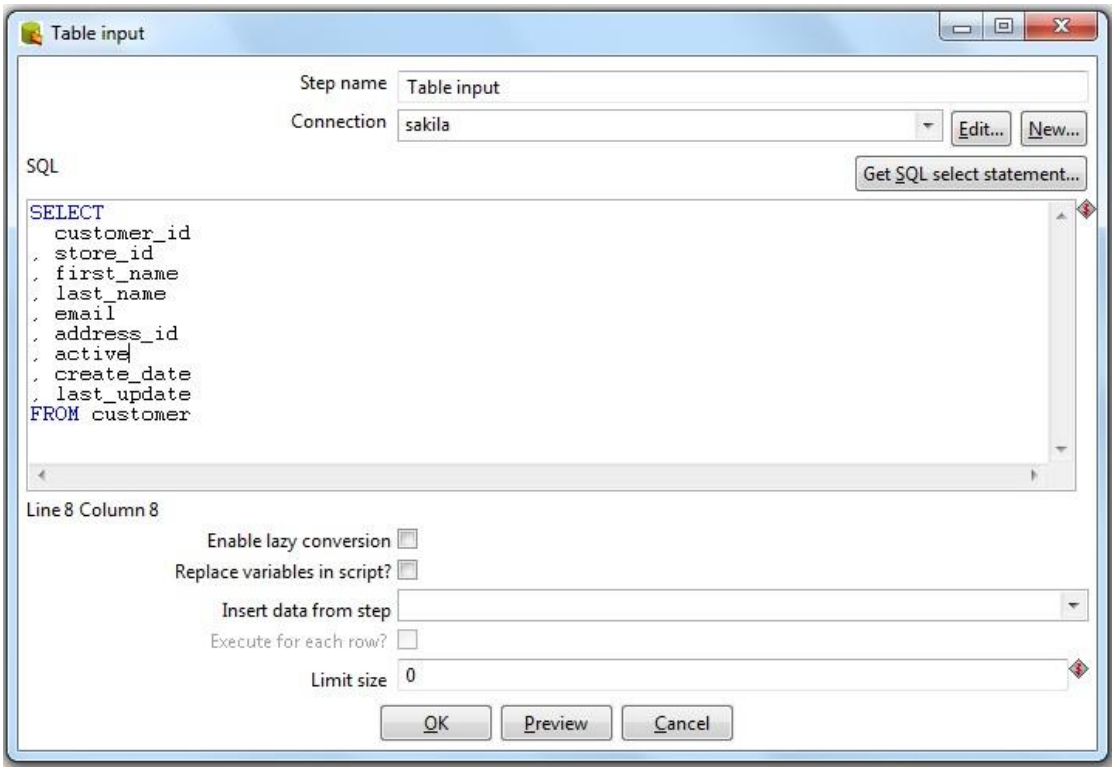
Pada Step ‘Table Input’, buat koneksi baru ke database sakila.



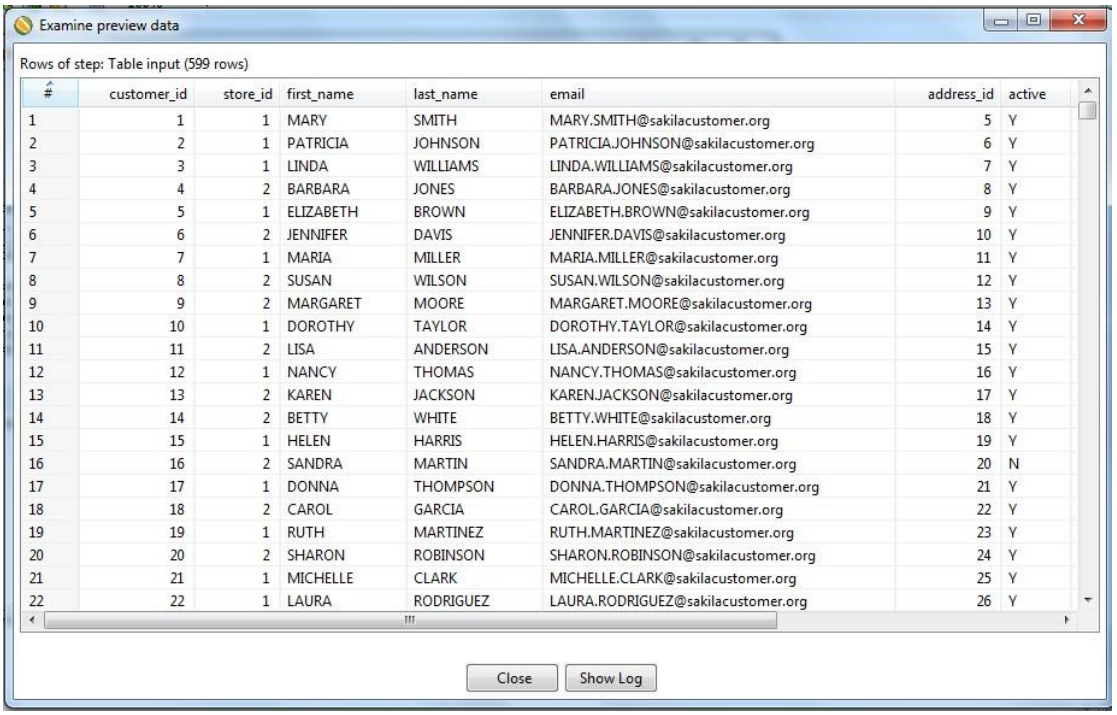
Lalu masukan sql untuk mengambil data dari tabel dan database yang kita pilih.



Akan tampil query sql yang dari pengambilan tabel yang kita pilih.

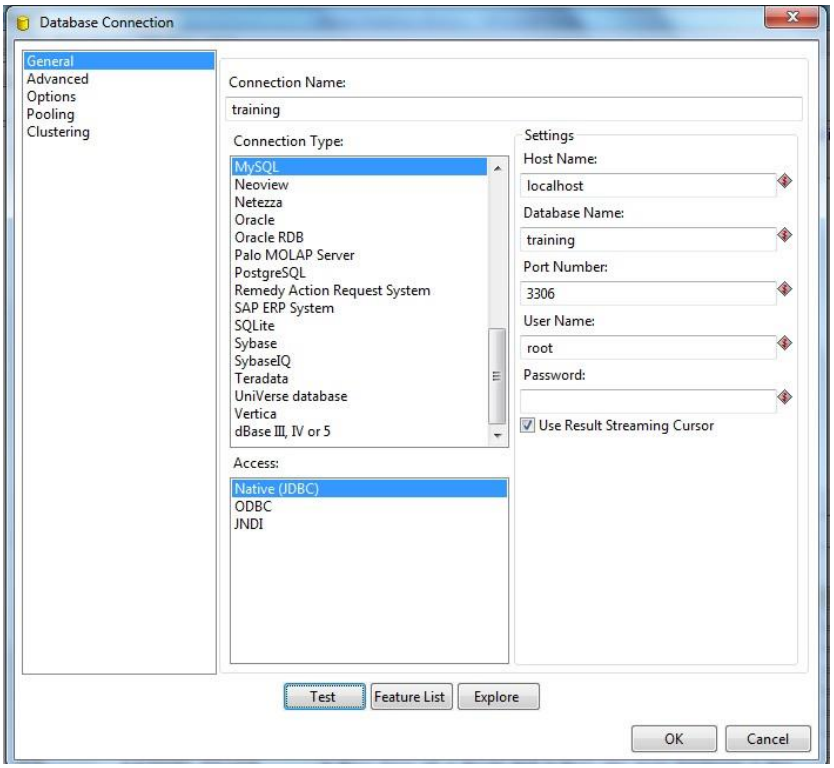


Jika kita klik preview, maka kan muncul preview hasil dari query tersebut.

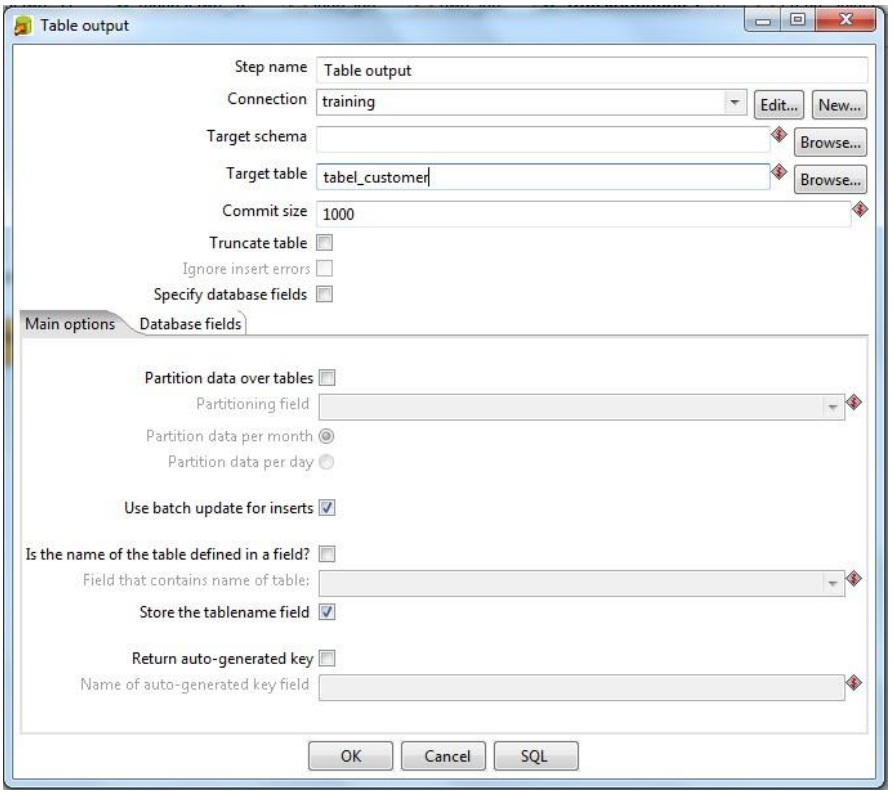


#	customer_id	store_id	first_name	last_name	email	address_id	active
1	1	1	MARY	SMITH	MARY.SMITH@sakilacustomer.org	5	Y
2	2	1	PATRICIA	JOHNSON	PATRICIA.JOHNSON@sakilacustomer.org	6	Y
3	3	1	LINDA	WILLIAMS	LINDA.WILLIAMS@sakilacustomer.org	7	Y
4	4	2	BARBARA	JONES	BARBARA.JONES@sakilacustomer.org	8	Y
5	5	1	ELIZABETH	BROWN	ELIZABETH.BROWN@sakilacustomer.org	9	Y
6	6	2	JENNIFER	DAVIS	JENNIFER.DAVIS@sakilacustomer.org	10	Y
7	7	1	MARIA	MILLER	MARIA.MILLER@sakilacustomer.org	11	Y
8	8	2	SUSAN	WILSON	SUSAN.WILSON@sakilacustomer.org	12	Y
9	9	2	MARGARET	MOORE	MARGARET.MOORE@sakilacustomer.org	13	Y
10	10	1	DOROTHY	TAYLOR	DOROTHY.TAYLOR@sakilacustomer.org	14	Y
11	11	2	LISA	ANDERSON	LISA.ANDERSON@sakilacustomer.org	15	Y
12	12	1	NANCY	THOMAS	NANCY.THOMAS@sakilacustomer.org	16	Y
13	13	2	KAREN	JACKSON	KAREN.JACKSON@sakilacustomer.org	17	Y
14	14	2	BETTY	WHITE	BETTY.WHITE@sakilacustomer.org	18	Y
15	15	1	HELEN	HARRIS	HELEN.HARRIS@sakilacustomer.org	19	Y
16	16	2	SANDRA	MARTIN	SANDRA.MARTIN@sakilacustomer.org	20	N
17	17	1	DONNA	THOMPSON	DONNA.THOMPSON@sakilacustomer.org	21	Y
18	18	2	CAROL	GARCIA	CAROL.GARCIA@sakilacustomer.org	22	Y
19	19	1	RUTH	MARTINEZ	RUTH.MARTINEZ@sakilacustomer.org	23	Y
20	20	2	SHARON	ROBINSON	SHARON.ROBINSON@sakilacustomer.org	24	Y
21	21	1	MICHELLE	CLARK	MICHELLE.CLARK@sakilacustomer.org	25	Y
22	22	1	LAURA	RODRIGUEZ	LAURA.RODRIGUEZ@sakilacustomer.org	26	Y

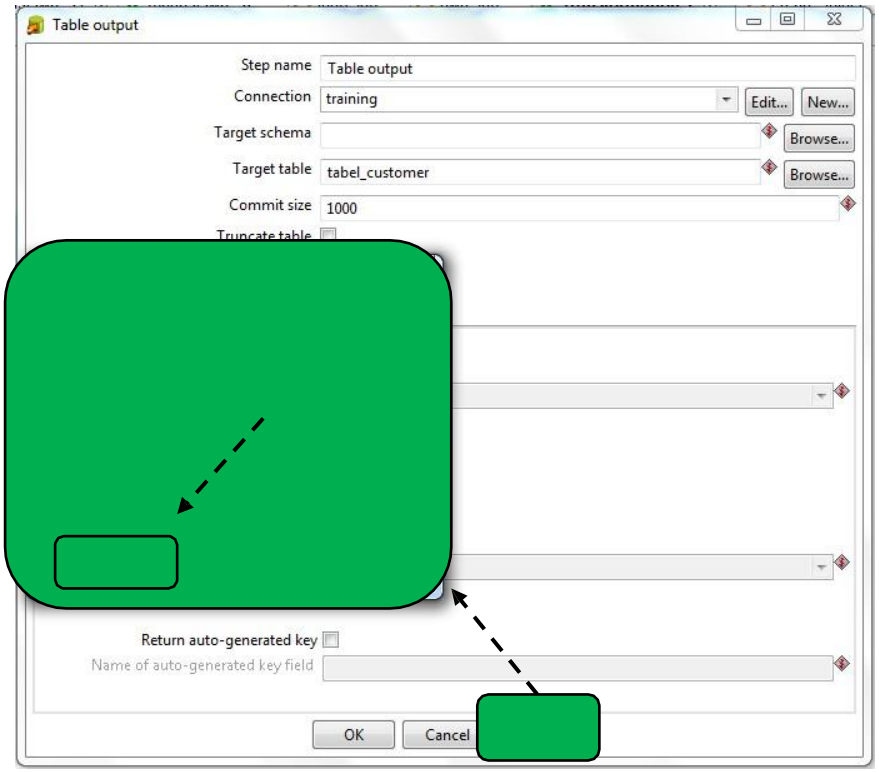
Setelah itu, masuk ke Step ‘Table Output’ dengan double click di icon tersebut. Buat koneksi ke database tujuan.



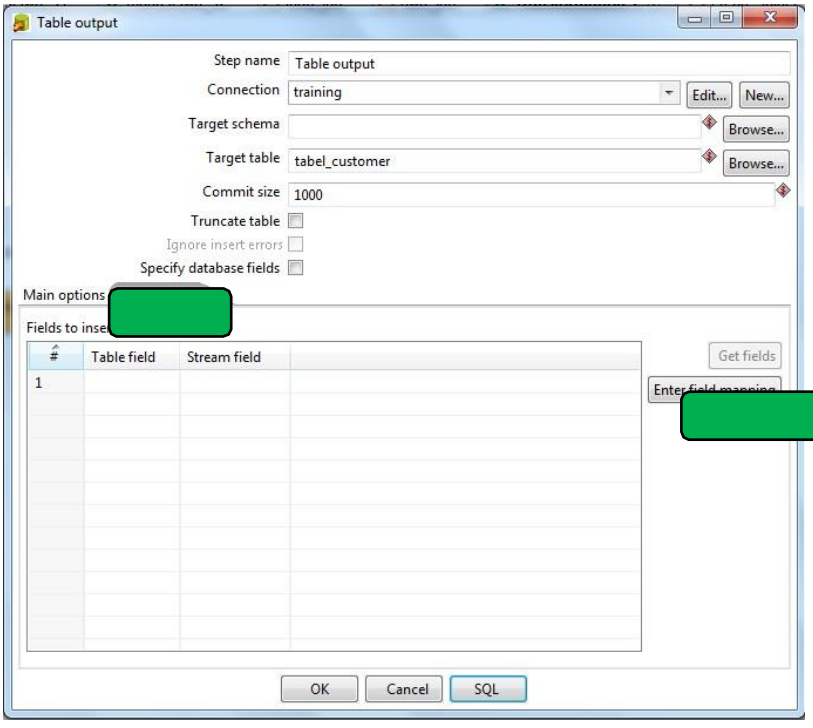
Tuliskan nama tabel pada field target tabel.



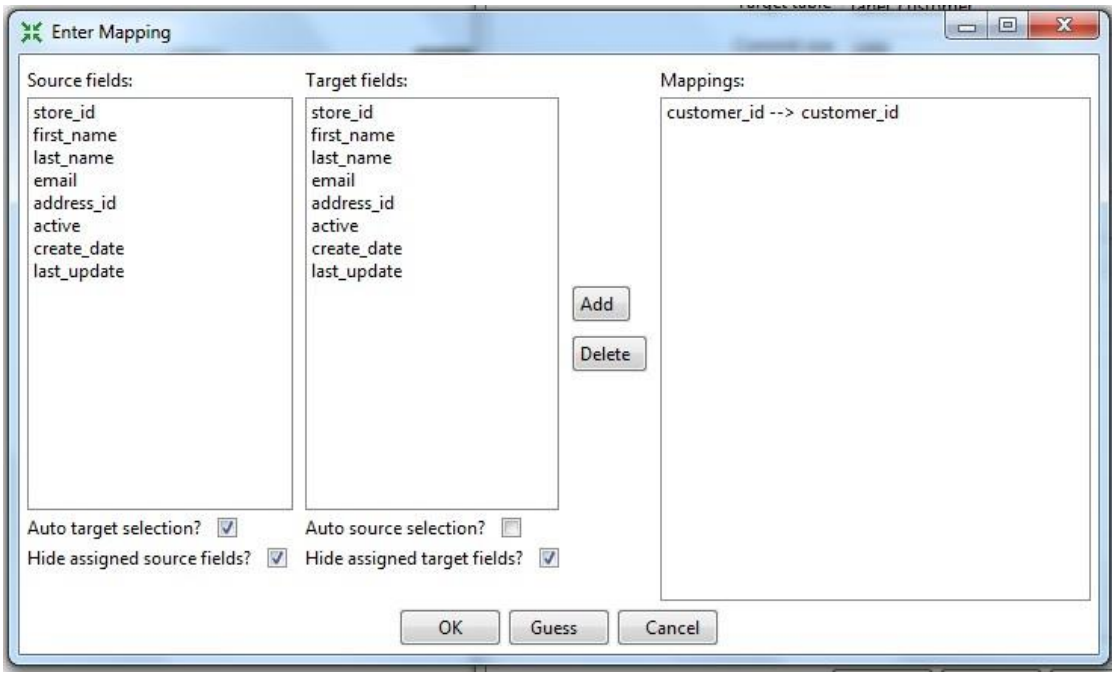
Seandainya tabel database tujuan belum ada, kita dapat meng-*generate*-nya dengan mengklik tombol SQL. Maka akan muncul ‘Simple SQL Editor’, klik tombol execute untuk menjalan sql tersebut. Secara otomatis pada database training akan terbangun tabel bernama ‘tabel_customer’.



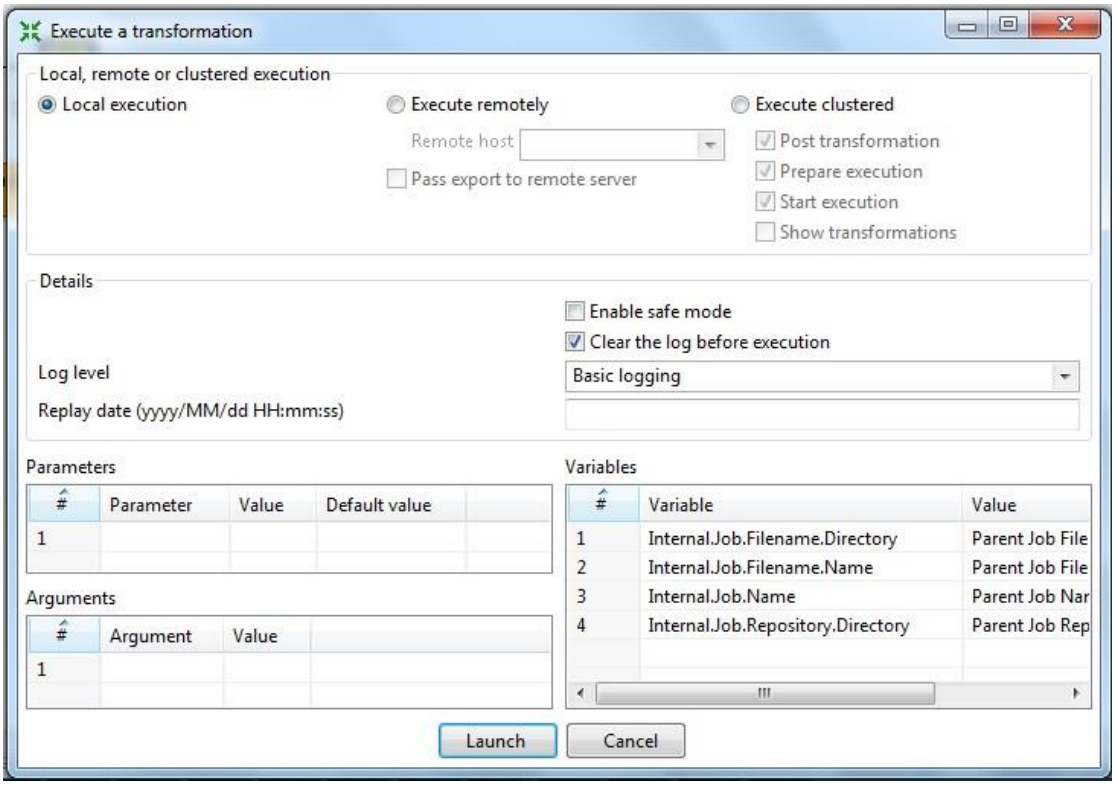
Masuk ke tab ‘Database fields’ dan klik tombol Enter field mapping untuk melakukan mapping antara data stream ke field yang tersedia.



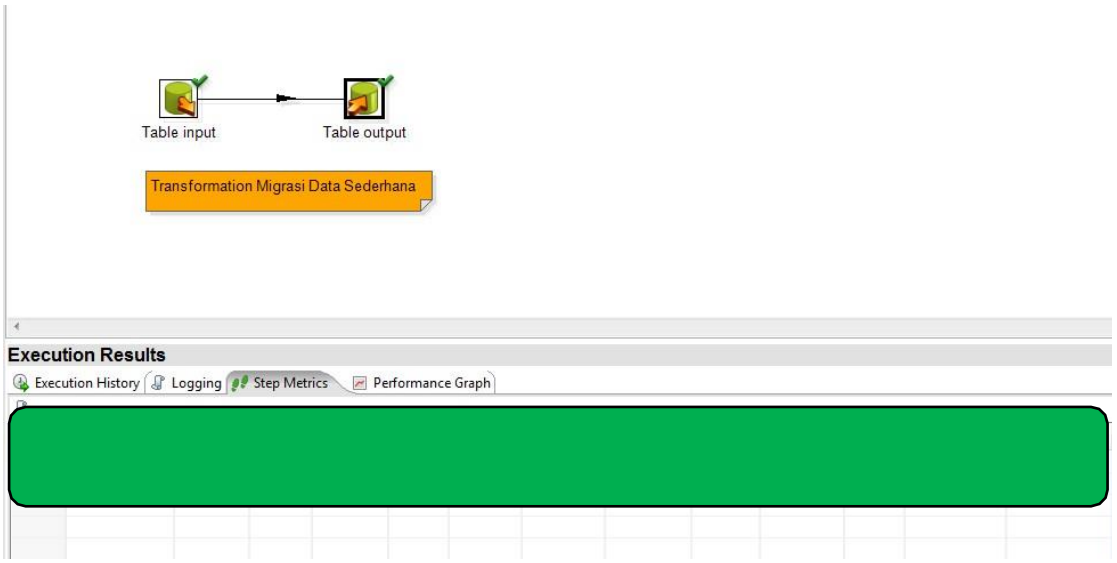
Lakukan mapping dengan mengklik item di ‘Source fields’ dan ‘Target Fields’ lalu klik tombol ‘Add’. Jika kita yakin nama field source dan target sama, kita dapat secara otomatis melakukan mapping dengan mengklik tombol ‘Guess’



Jalankan transformation dengan tombol F9 atau mengklik tombol  lalu pada kotak dialog konfirmasi klik ‘Launch’



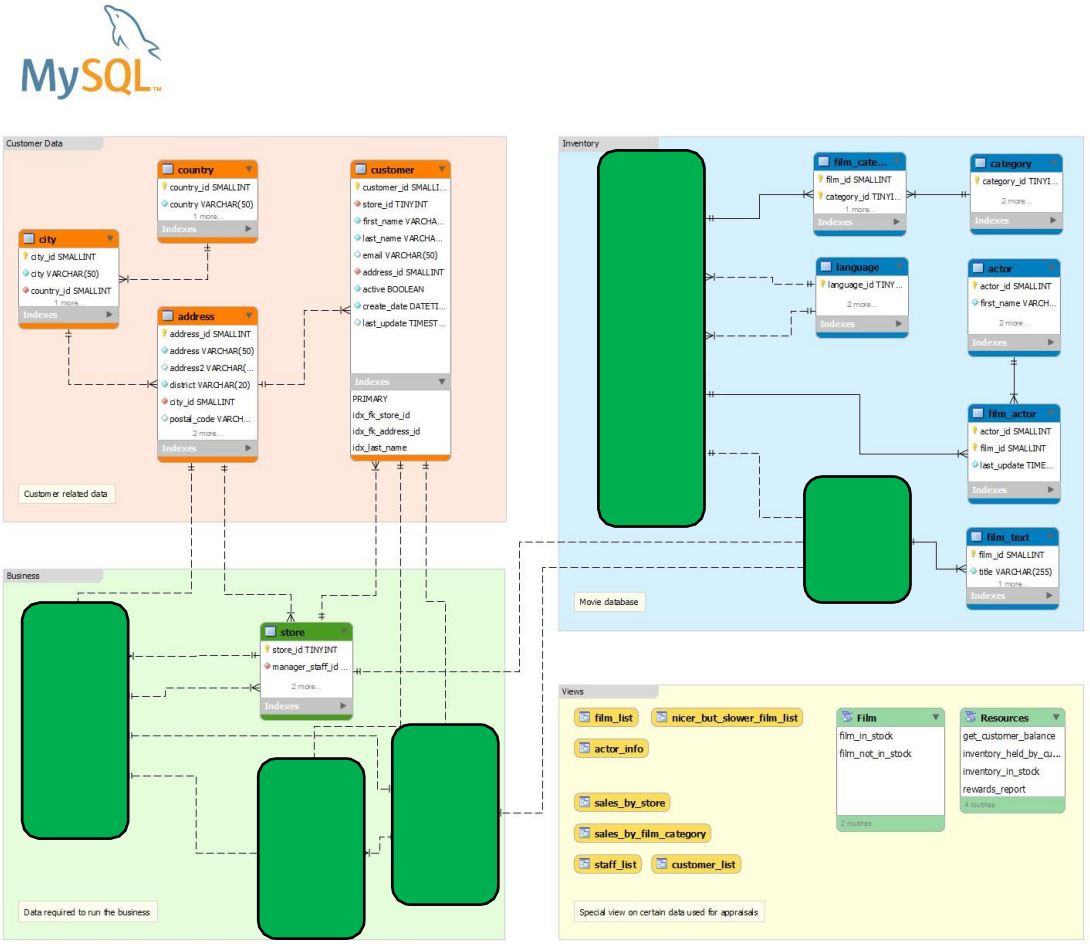
Jika sukses maka akan muncul tampilan seperti berikut



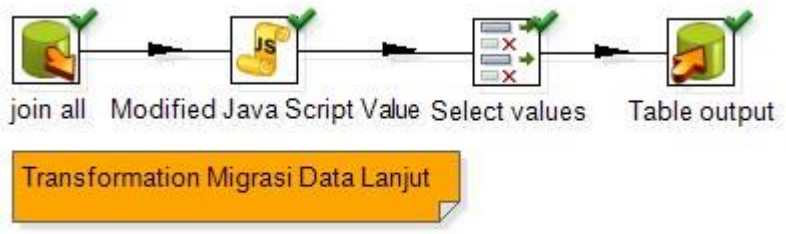
Dari Step Metrics terlihat bahwa ada baris/row data sebanyak 599 yang dimigrasi.

4.4.2.4 Migrasi Data Lanjut

Kita akan membuat sebuah tabel yang berisi data yang berasal dari tabel- tabel tertentu. Biasanya dilakukan untuk menganalisis sebuah transaksi data. Kasus yang diambil adalah kita akan membuat tabel yang merangkum dari tabel film, inventory, rental, dan payment.



Berikut langkah-langkahnya :



1. Pada step ‘join all’, field koneksi diisi dengan sakila. Sedangkan SQL nya diisi dengan query join semua tabel terkait.

```
SELECT *,year(payment_date) as tahun_pembayaran,
monthname(payment_date) as bulan_pembayaran, day(payment_date) as tanggal_pembayaran, year(rental_date) as tahun_pinjam, monthname(rental_date) as bulan_pinjam, day(rental_date) as tanggal_pinjam, year(return_date) as tahun_kembali, monthname(return_date) as bulan_kembali, day(return_date) as tanggal_kembali
FROM film f JOIN inventory i ON f.film_id=i.film_id
JOIN rental r ON r.inventory_id=i.inventory_id
JOIN payment p ON p.rental_id=r.rental_id
```

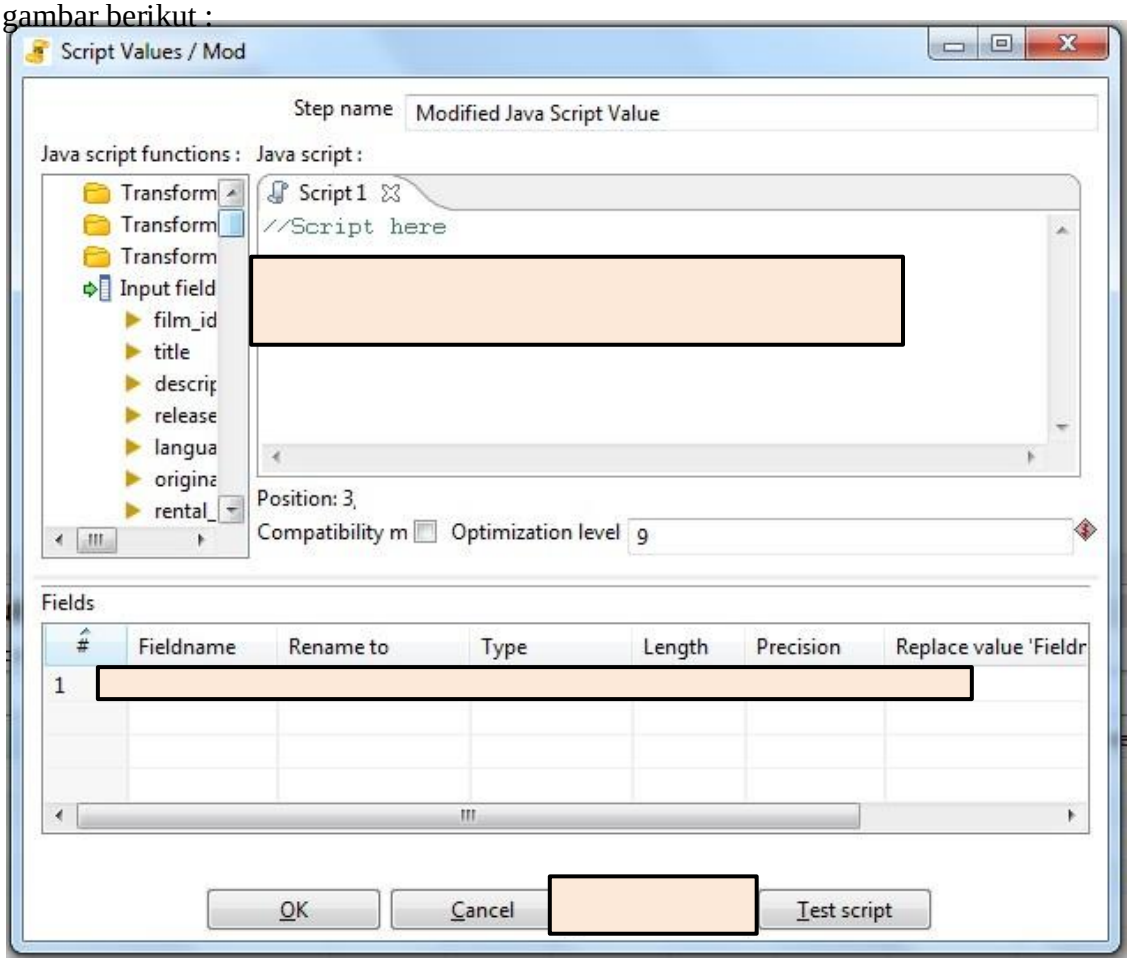
2. Pada Step Modified Java Script Value diisi dengan sintaks seperti berikut :

```
var nama_depan=first_name;

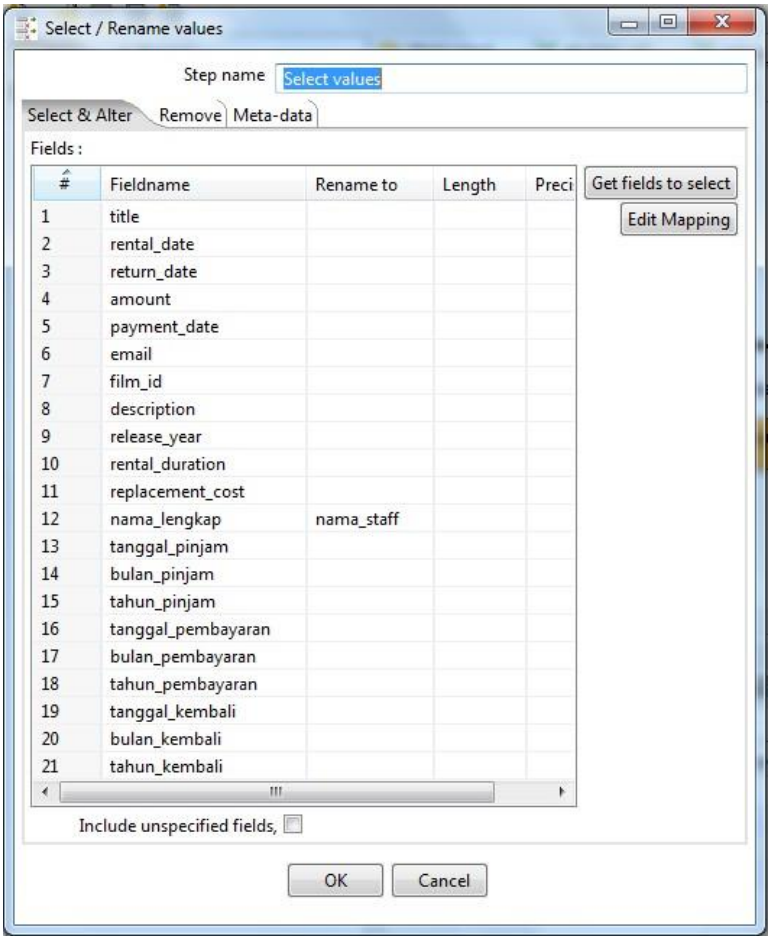
var nama_belakang=last_name;

var result = nama_depan +" "+ nama_belakang;
```

Maksud dari sintaks di atas adalah untuk menggabung (concat) antara field first_name dengan last_name lalu dimasukan ke dalam variable result. Setelah itu dilakukan get field dari sintaks di atas dengan mengklik tombol Get variables seperti gambar berikut :

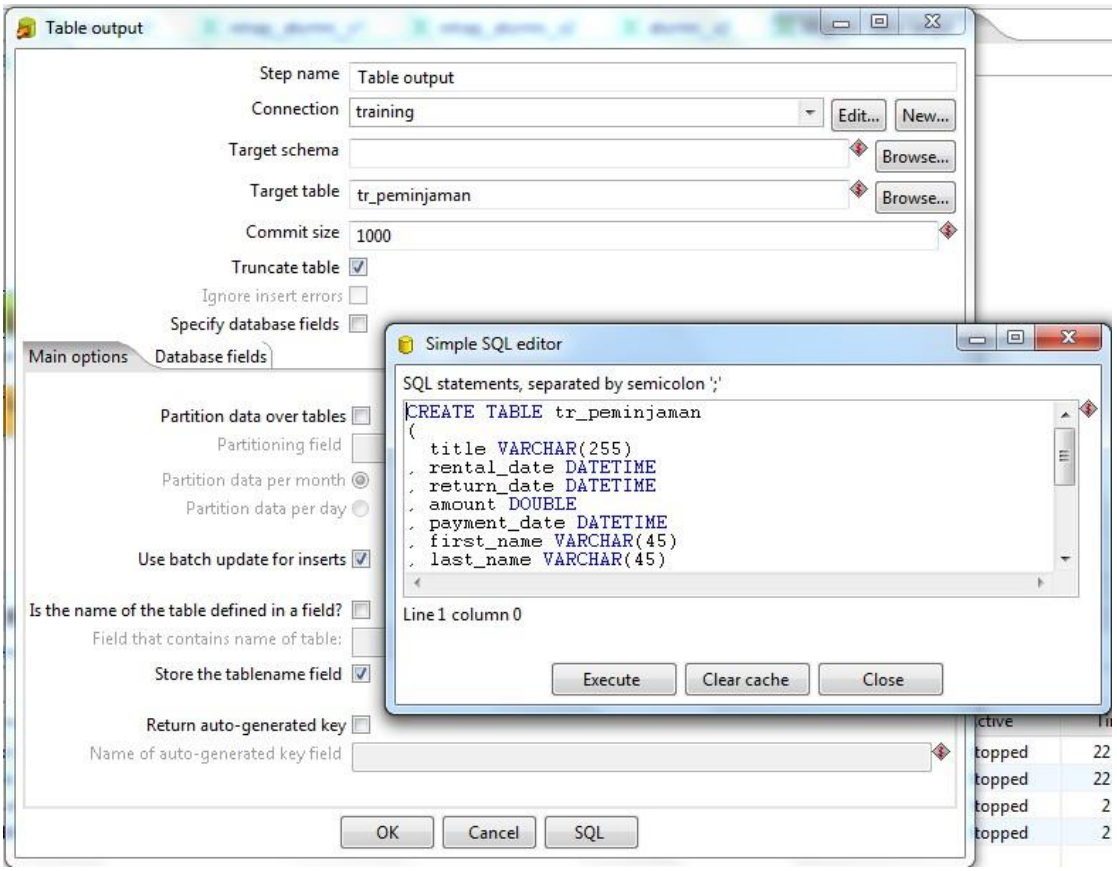


3. Pada step ‘Select Value’, diisikan seperti berikut



#	Fieldname	Rename to	Length	Preci
1	title			
2	rental_date			
3	return_date			
4	amount			
5	payment_date			
6	email			
7	film_id			
8	description			
9	release_year			
10	rental_duration			
11	replacement_cost			
12	nama_lengkap	nama_staff		
13	tanggal_pinjam			
14	bulan_pinjam			
15	tahun_pinjam			
16	tanggal_pembayaran			
17	bulan_pembayaran			
18	tahun_pembayaran			
19	tanggal_kembali			
20	bulan_kembali			
21	tahun_kembali			

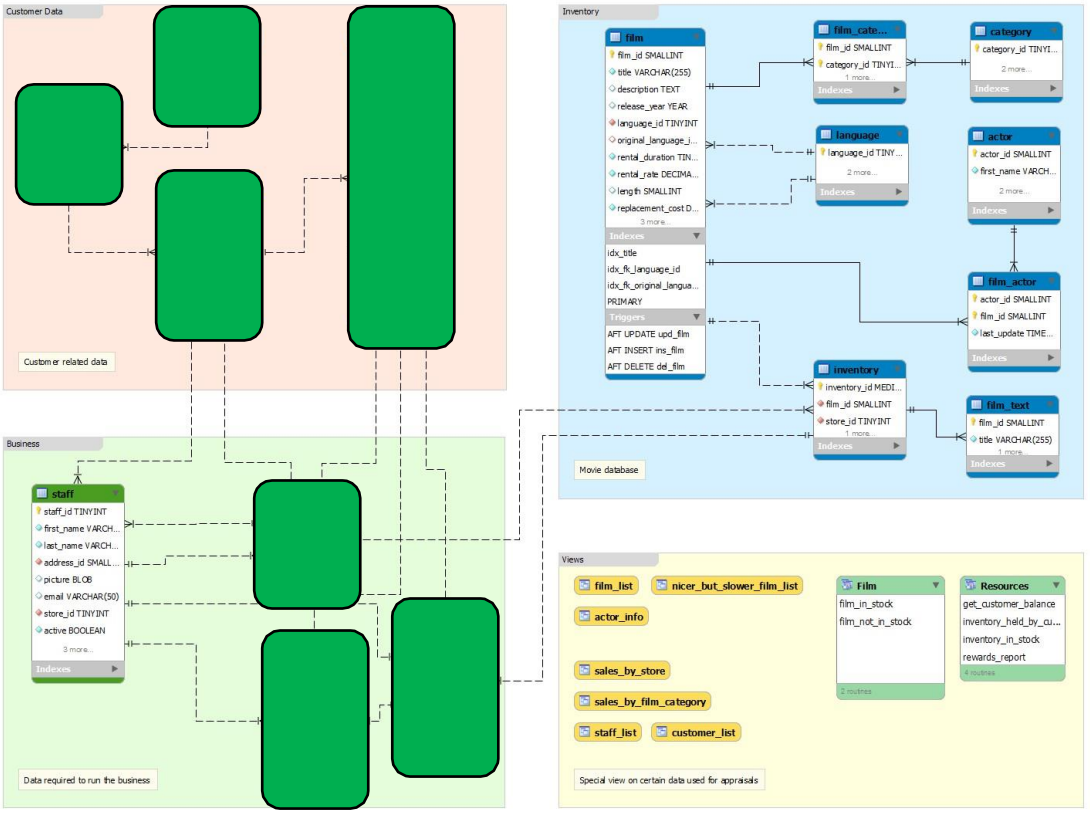
4. Pada Step ‘Table Output’, isi Connection dengan koneksi ke database yang kita gunakan, yaitu dbms ‘MySQL, ’ database ‘training’, hostname ‘localhost’. Jika tabel belum dibuat, klik tombol SQL untuk menggenerate SQL lalu di execute.

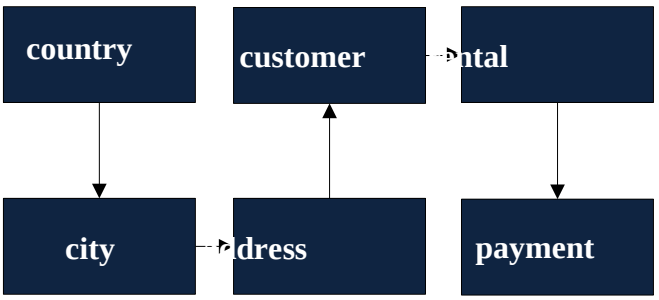


Setelah itu jalankan transformation dengan klik F5 atau tombol run ( ). Maka akan terbentuk tabel baru yang berisi dari join tabel yang dibutuhkan. Di section selanjutnya akan
dibahas penggunaannya di dalam OLAP, dan reporting.

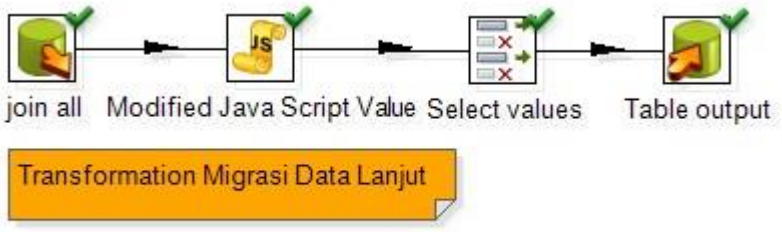
4.4.2.5 Migrasi Data Lanjut 2

Pada tahap ini kita akan mencoba membuat sebuah tabel yang akan dijadikan cube dengan tujuan kita dapat mengenali toko mana yang paling banyak transaksinya, daerah mana yang paling banyak transaksinya, customer mana yang paling banyak memberikan pemasukan
bagi perusahaan. Berikut adalah tabel yang akan kita gunakan.





Berikut langkah-langkahnya :



1. Pada step ‘join all’, field koneksi diisi dengan sakila. Sedangkan SQL nya diisi dengan query join semua tabel terkait.

```
SELECT *,
        YEAR(payment_date) AS tahun_pembayaran,
        MONTHNAME(payment_date) AS
        bulan_pembayaran, DAY(payment_date) AS
        tanggal_pembayaran, YEAR(rental_date) AS
        tahun_pinjam, MONTHNAME(rental_date) AS
        bulan_pinjam, DAY(rental_date) AS
        tanggal_pinjam, YEAR(return_date) AS
        tahun_kembali, MONTHNAME(return_date) AS
        bulan_kembali, DAY(return_date) AS
        tanggal_kembali
FROM customer a
JOIN address b ON  a.address_id=b.address_id
JOIN city c ON c.city_id=b.city_id
JOIN country d ON d.country_id=c.country_id
JOIN rental e ON e.customer_id=a.customer_id
JOIN payment f ON f.rental_id=e.rental_id
```

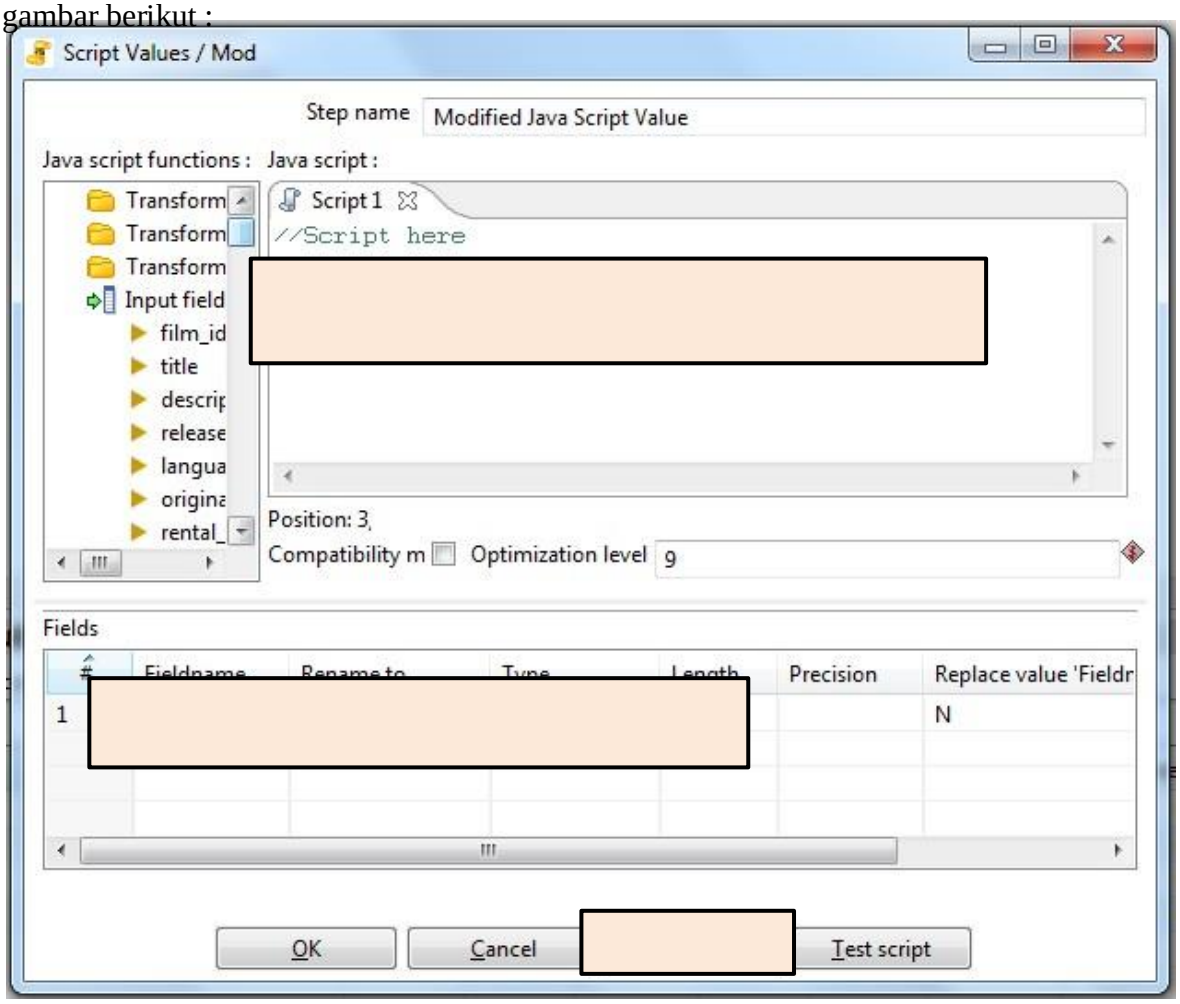
2. Pada Step Modified Java Script Value diisi dengan sintaks seperti berikut :

```
var nama_depan=first_name;

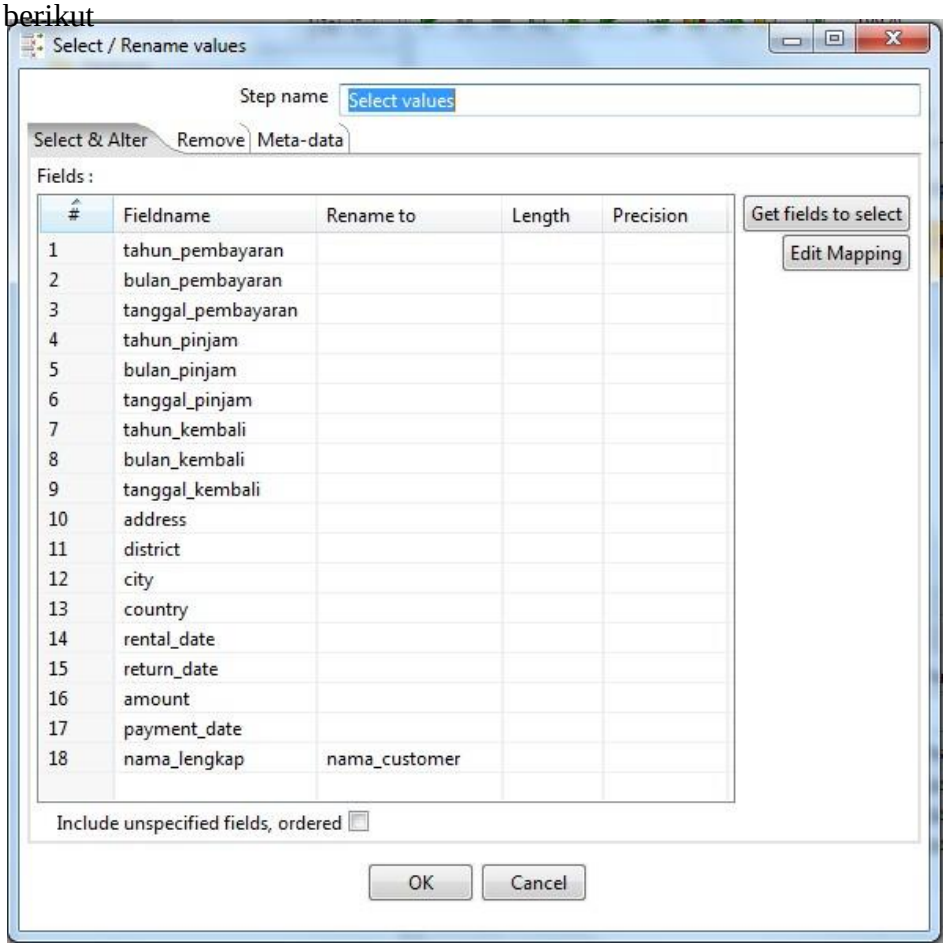
var nama_belakang=last_name;

var result = nama_depan +" "+ nama_belakang;
```

Maksud dari sintaks di atas adalah untuk menggabung (concat) antara field first_name dengan last_name lalu dimasukan ke dalam variable result. Setelah itu dilakukan get field dari sintaks di atas dengan mengklik tombol Get variables seperti gambar berikut :

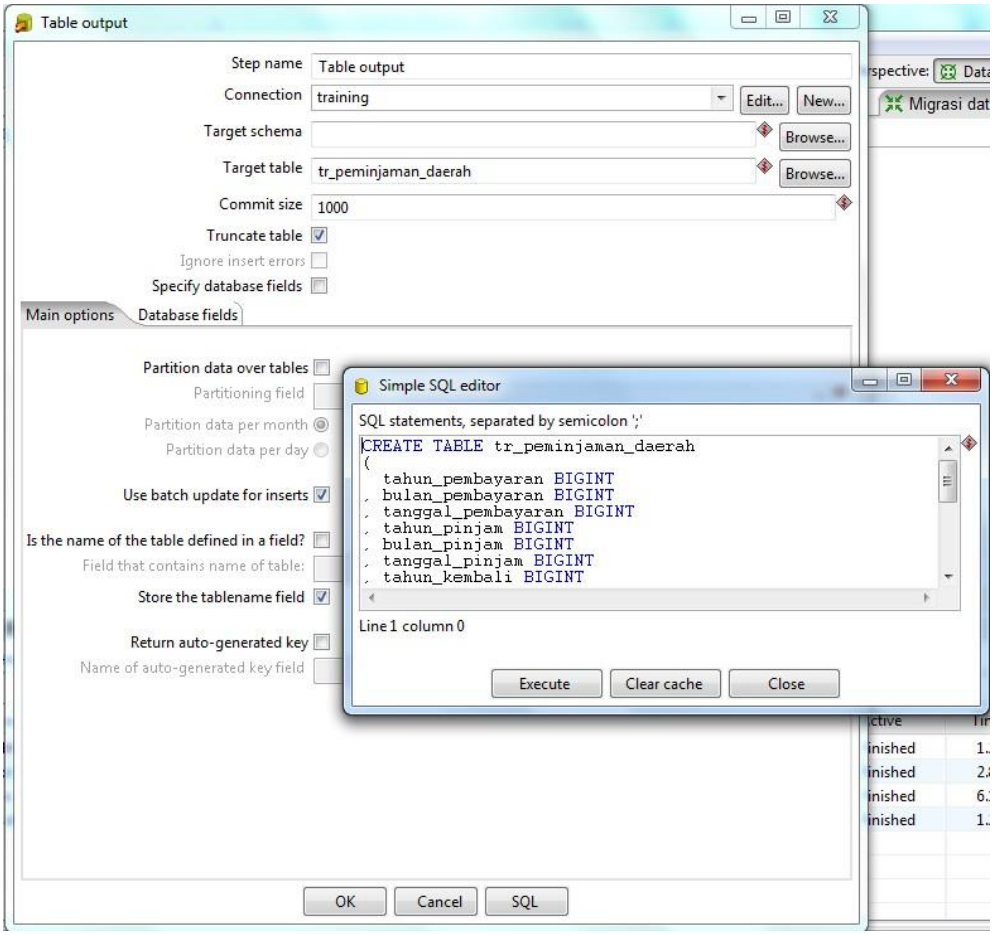


3. Pada step ‘Select Value’, diisikan seperti berikut



#	Fieldname	Rename to	Length	Precision
1	tahun_pembayaran			
2	bulan_pembayaran			
3	tanggal_pembayaran			
4	tahun_pinjam			
5	bulan_pinjam			
6	tanggal_pinjam			
7	tahun_kembali			
8	bulan_kembali			
9	tanggal_kembali			
10	address			
11	district			
12	city			
13	country			
14	rental_date			
15	return_date			
16	amount			
17	payment_date			
18	nama_lengkap	nama_customer		

4. Pada Step ‘Table Output’, isi Connection dengan koneksi ke database yang kita gunakan, yaitu dbms ‘MySQL, ’ database ‘training’, hostname ‘localhost’. Jika tabel belum dibuat, klik tombol SQL untuk menggenerate SQL lalu di execute.



Setelah itu jalankan transformation dengan klik F5 atau tombol run ( ). Maka akan terbentuk tabel baru yang berisi dari join tabel yang dibutuhkan. Di section selanjutnya akan dibahas penggunaannya di dalam OLAP, dan reporting.

4.4.3 Penggunaan Pan dan Kitchen Dalam Otomasi Transformation dan Job

Pada pembahasan sebelumnya disebutkan bahwa Pan digunakan untuk menjalan transformation sedangkan Kitchen digunakan dalam menjelankan job via command line. Berikut contoh penggunaan sederhana Pan :

- Menjalankan file transformation : Migrasi_data_lanjut.ktr

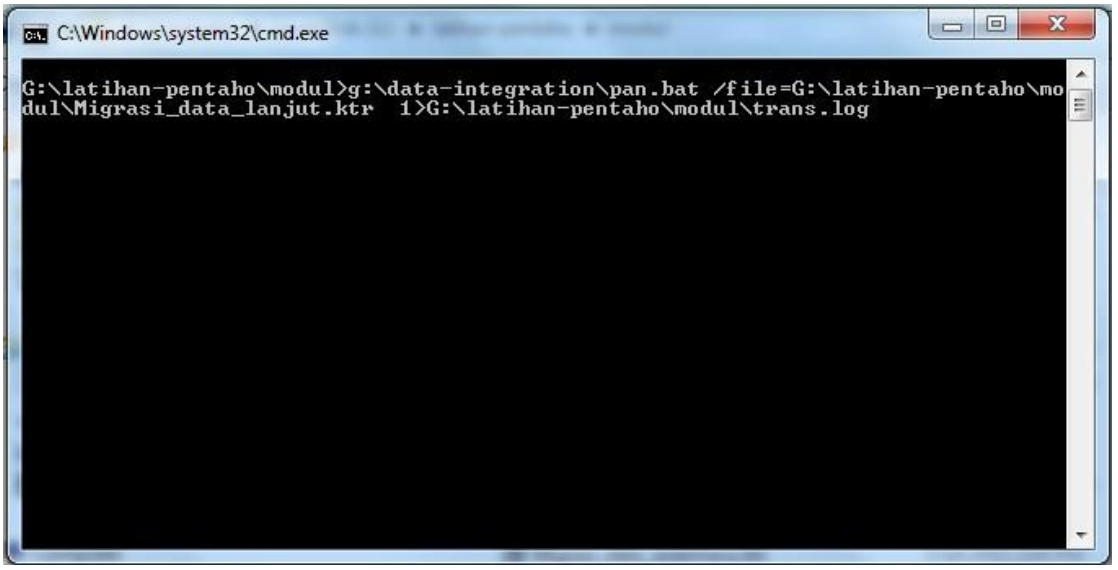
  Pada notepad ++, buatlah file berekstensi .bat yang isinya :

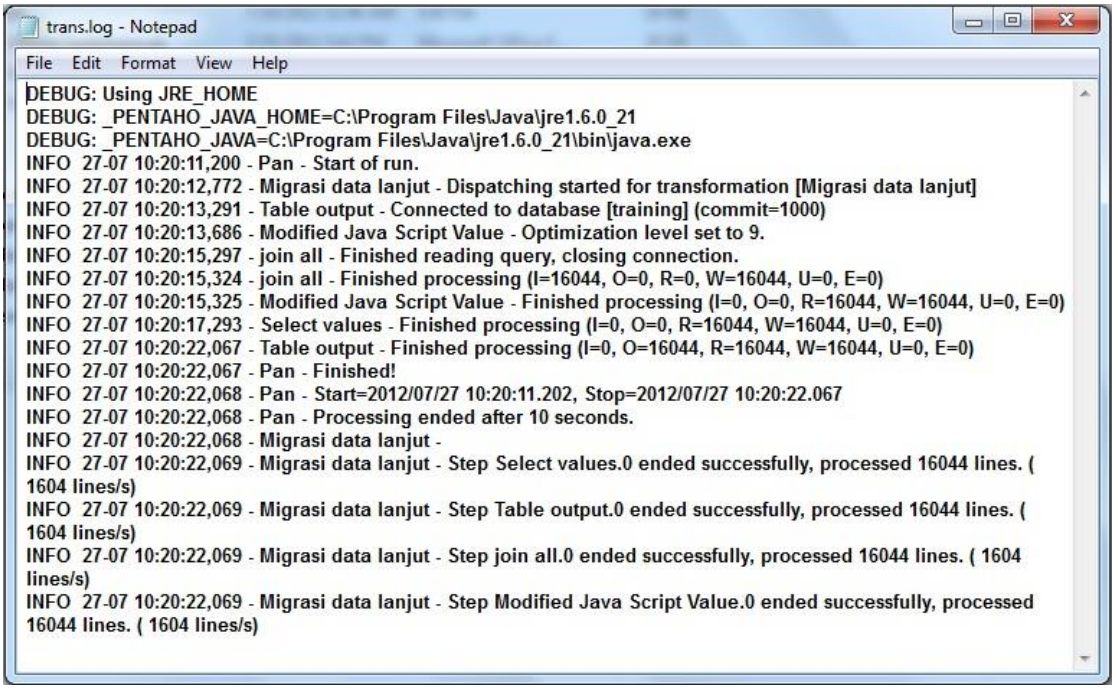
[path file pan.bat]\pan.bat /file=[path dan nama file transformation].ktr > [file_log untuk menyimpan log hasil proses pan].log

Contohnya :

g:\data-integration\pan.bat /file=G:\latihan-pentaho\modul\ Migrasi_data_lanjut.ktr > G:\latihan-pentaho\modul\trans.log

untuk menjalankannya, double klik pada file .bat yang telah kita buat. Setelah itu kita cek pada file log yang bernama trans.log.





- Menjalankan file job : email1.kjb

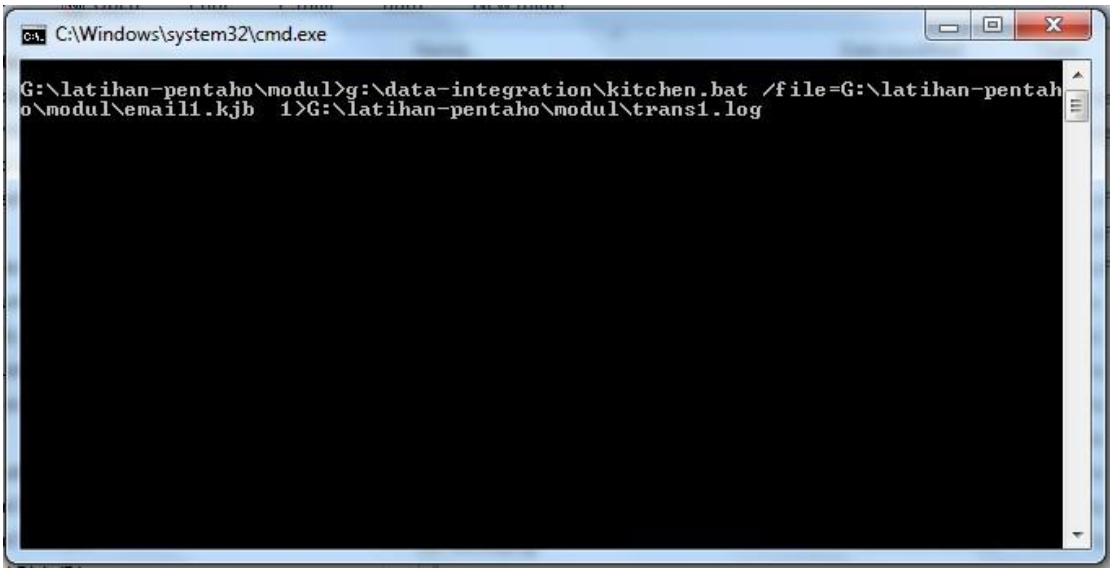
  Pada notepad ++, buatlah file berekstensi .bat yang isinya :

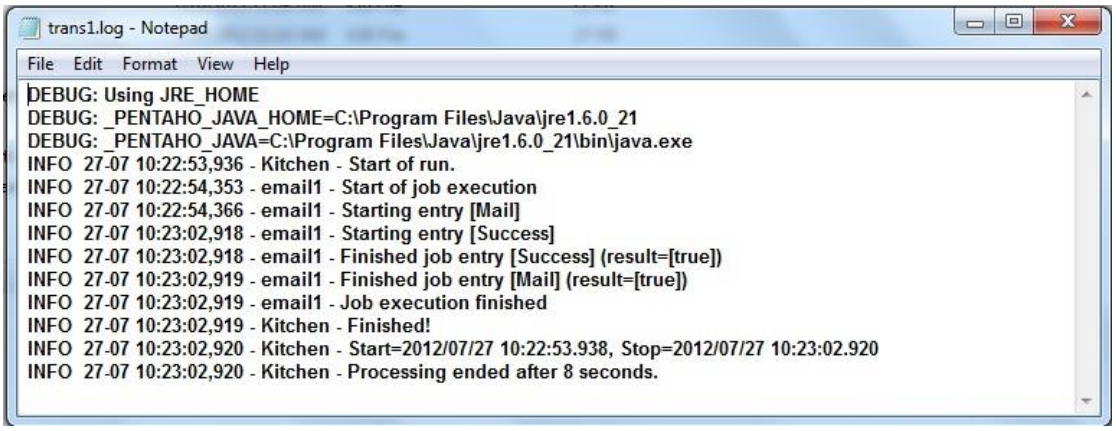
  [path file kitchen.bat]\kitchen.bat /file=[path dan nama file job].ktr > [file_log untuk menyimpan log hasil proses kitchen].log

  Contohnya :

  g:\data-integration\kitchen.bat /file=G:\latihan-pentaho\modul\ email1.kjb > G:\latihan-pentaho\modul\trans1.log

  untuk menjalankannya, double klik pada file .bat yang telah kita buat. Setelah itu kita cek pada file log yang bernama trans1.log.



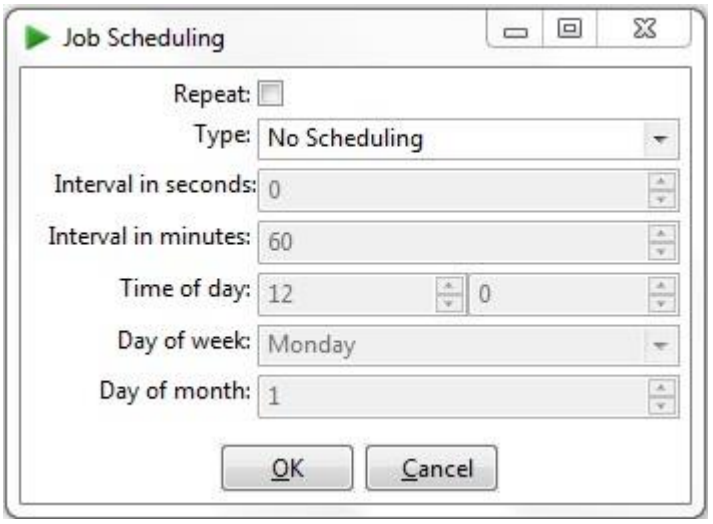


Setelah kita memahami konsep Job dan Transformasi juga cara menjalankan Pan dan Kitchen, kita akan melakukan otomasi penjalanan aplikasi tersebut (otomasi/schedulling). Ada beberapa cara melakukan scheduling, yaitu menggunakan Spoon (GUI) dan

menggunakan Pan/Kitchen dibantu dengan Windows Task Scheduller (untuk Linux menggunakan Crontab).

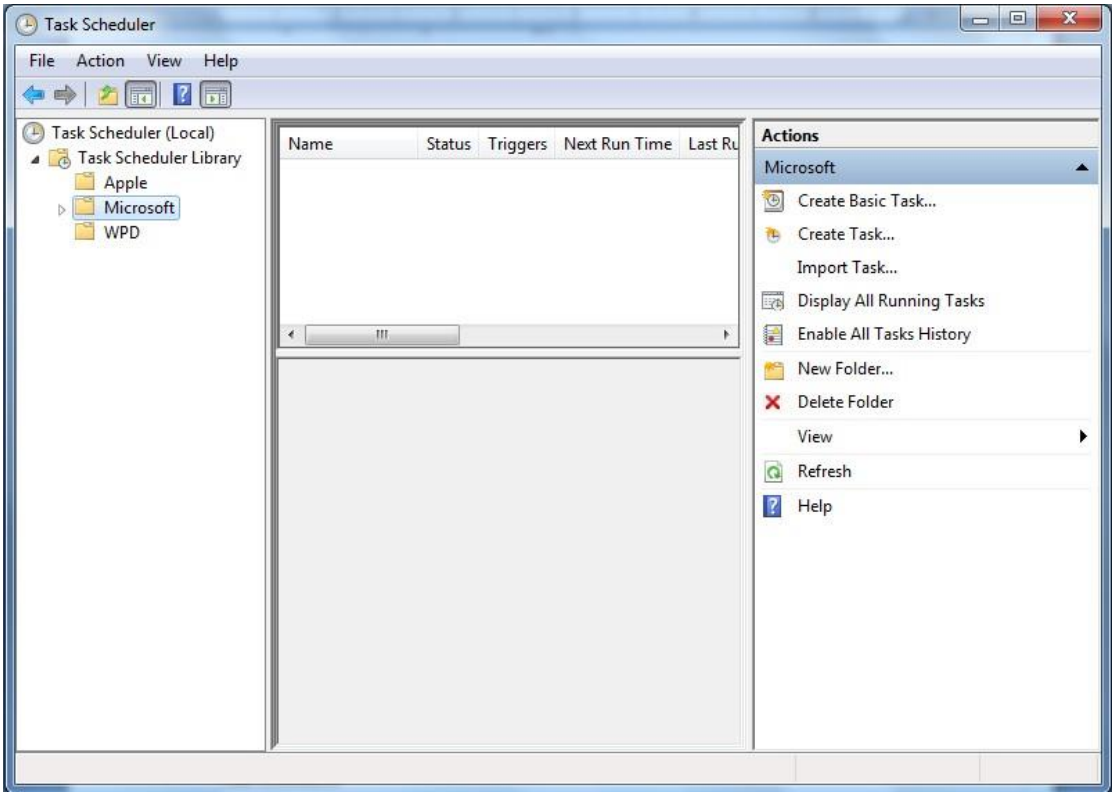
Menggunakan Spoon

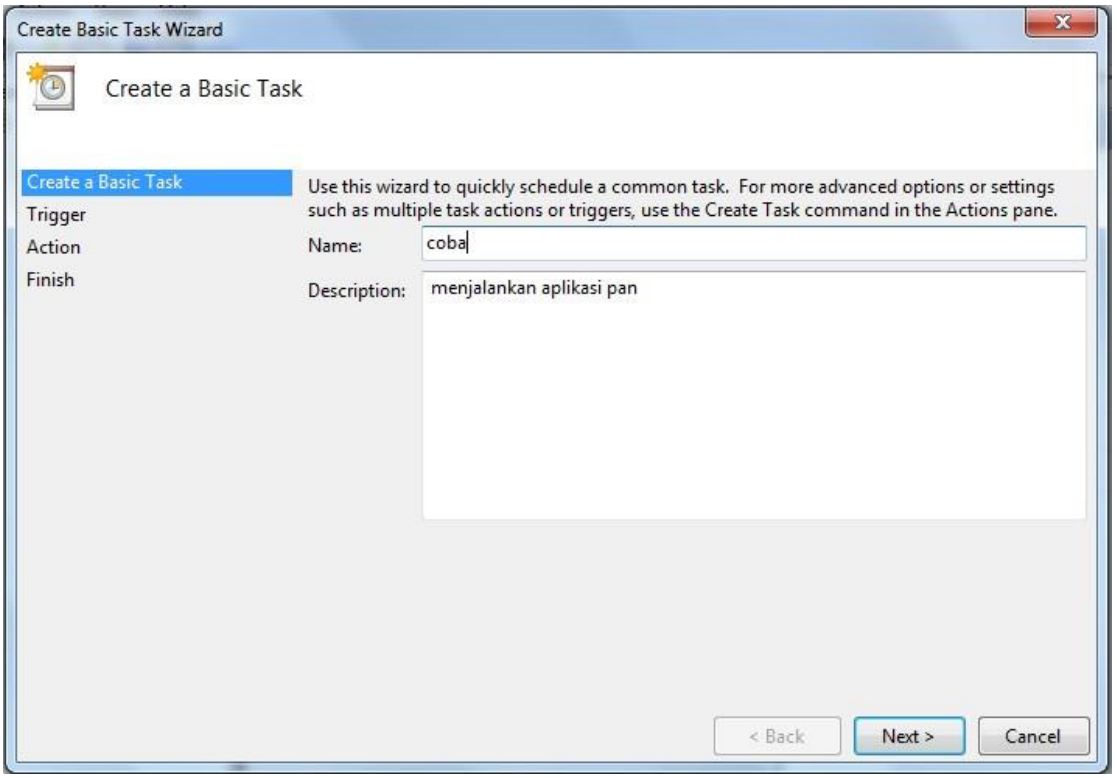
Pada Step ‘START’, kita dapat mensetting penjadwalan sesuai kebutuhan.

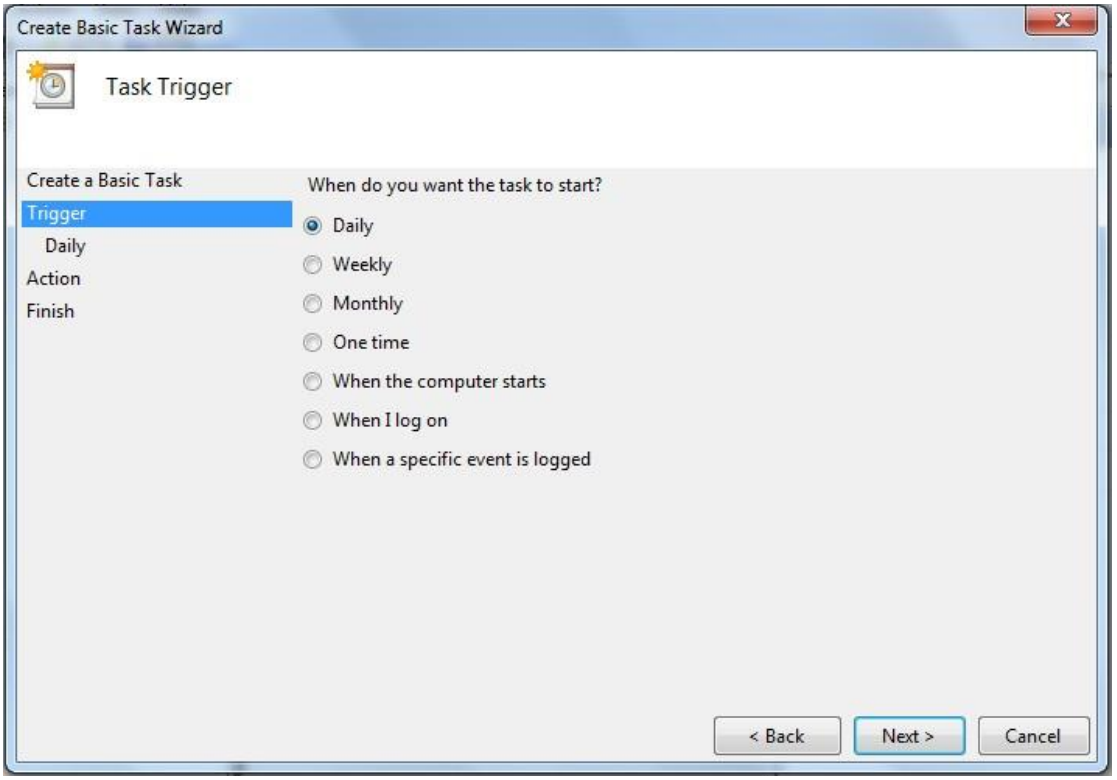


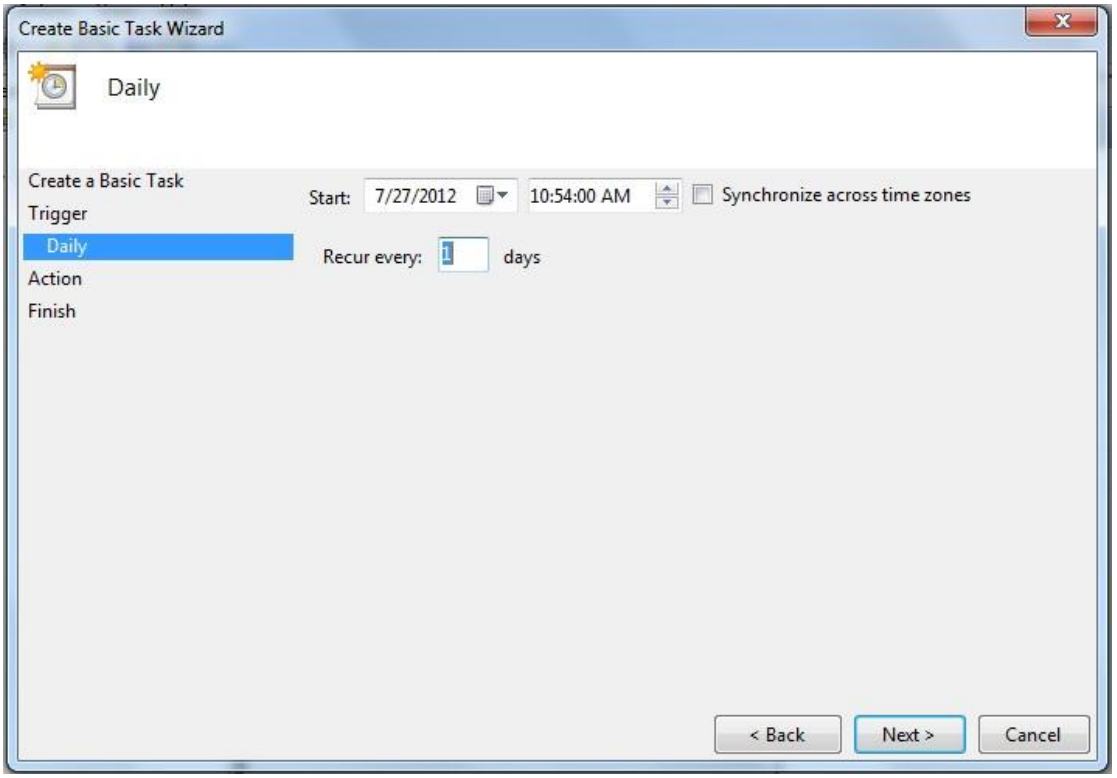
Menggunakan Pan/Kitchen

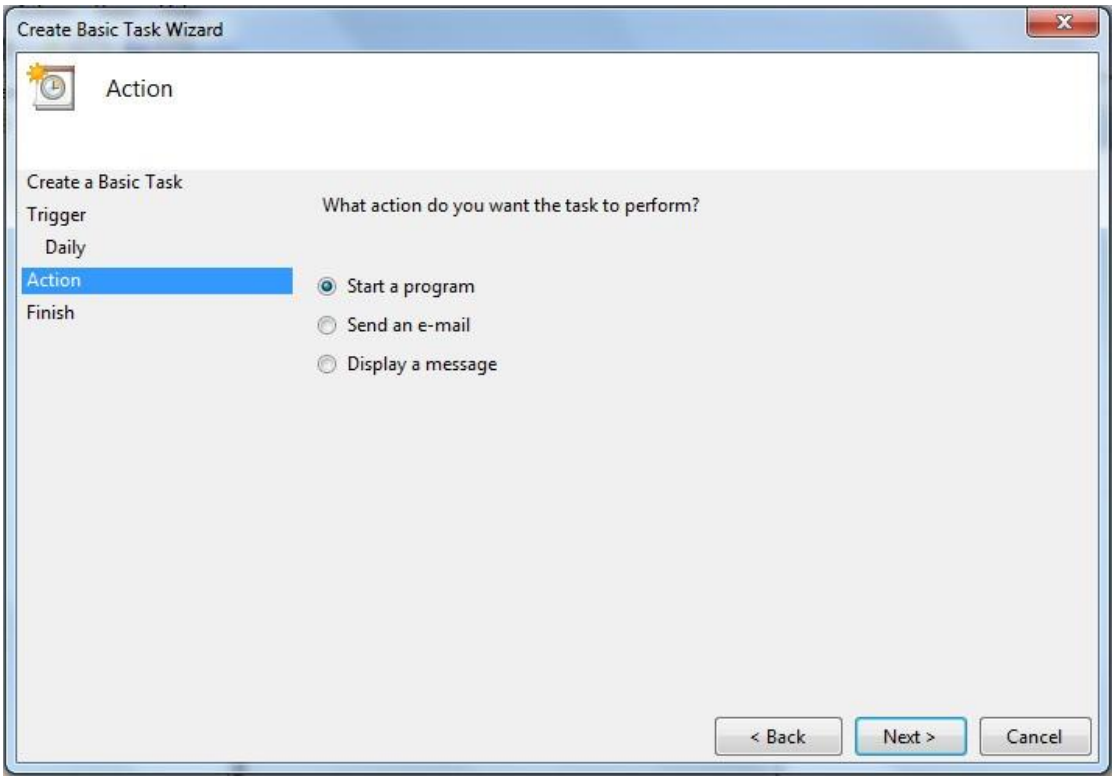
Setelah dibuat file .bat tersebut, kita masuk ke aplikasi bawaan windows, yaitu Windows Task Scheduller yang berada pada Control Panel Windows.

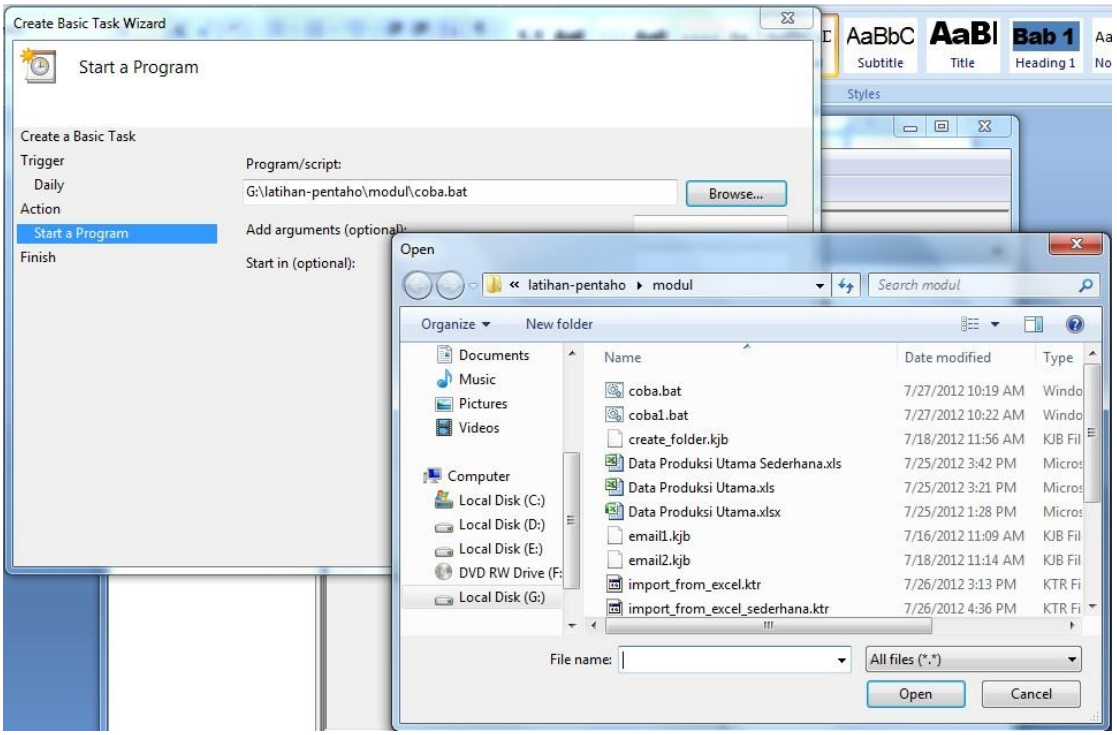


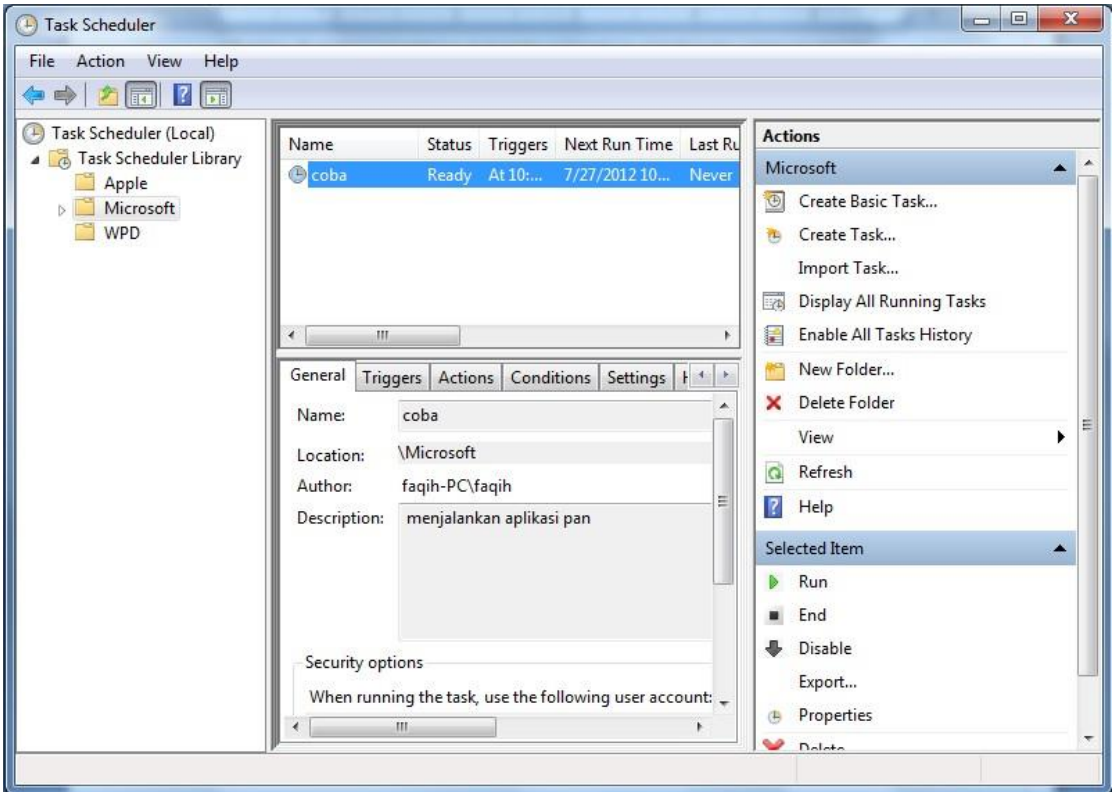

Create Basic Task Wizard
Create a Basic Task
Create a Basic Task
Trigger
Action
Finish
Use this wizard to quickly schedule a common task. For more advanced options or settings such as multiple task actions or triggers, use the Create Task command in the Actions pane.
Name:
coba
Description:
menjalankan aplikasi pan
< Back
Next >
Cancel


Create Basic Task Wizard
Task Trigger
Create a Basic Task
Trigger
Daily
Action
Finish
When do you want the task to start?
Daily
Weekly
Monthly
One time
When the computer starts
When I log on
When a specific event is logged
< Back
Next >
Cancel


Create Basic Task Wizard
Daily
Create a Basic Task
Trigger
Daily
Action
Finish
Start: 7/27/2012 10:54:00 AM
Synchronize across time zones
Recur every: 1 days
< Back
Next >
Cancel


Create Basic Task Wizard
Action
Create a Basic Task
Trigger
Daily
Action
Finish
What action do you want the task to perform?
Start a program
Send an e-mail
Display a message
< Back
Next >
Cancel


Create Basic Task Wizard
Start a Program
Create a Basic Task
Trigger
Daily
Action
Start a Program
Finish
Program/script:
G:\latihan-pentaho\modul\coba.bat
Browse...
Add arguments (optional):
Start in (optional):
Open
latihan-pentaho ▸ modul
Search modul
Organize
New folder
Documents
Music
Pictures
Videos
Computer
Local Disk (C:)
Local Disk (D:)
Local Disk (E:)
DVD RW Drive (F:
Local Disk (G:)
Name
Date modified
Type
coba.bat 7/27/2012 10:19 AM
coba1.bat 7/27/2012 10:22 AM
create_folder.kjb 7/18/2012 11:56 AM
Data Produksi Utama Sederhana.xls 7/25/2012 3:42 PM
Data Produksi Utama.xls 7/25/2012 3:21 PM
Data Produksi Utama.xlsx 7/25/2012 1:28 PM
email1.kjb 7/16/2012 11:09 AM
email2.kjb 7/18/2012 11:14 AM
import_from_excel.ktr 7/26/2012 3:13 PM
import_from_excel_sederhana.ktr 7/26/2012 4:36 PM
File name:
All files (*.*)
Open
Cancel
AaBbC
Subtitle
AaBl
Title
Bab 1
Heading 1
Styles


Task Scheduler
File
Action
View
Help
Task Scheduler (Local)
Task Scheduler Library
Apple
Microsoft
WPD
Name
Status
Triggers
Next Run Time
Last Ru
coba
Ready
At 10:...
7/27/2012 10...
Never
General
Triggers
Actions
Conditions
Settings
Name:
coba
Location:
\Microsoft
Author:
faqih-PC\faqih
Description:
menjalankan aplikasi pan
Security options
When running the task, use the following user account:
Actions
Microsoft
Create Basic Task...
Create Task...
Import Task...
Display All Running Tasks
Enable All Tasks History
New Folder...
Delete Folder
View
Refresh
Help
Selected Item
Run
End
Disable
Export...
Properties

Bab 5 Pentaho Report Designer

Pentaho Report Designer adalah tools untuk merancang laporan. Hasil dari racangan tersebut lalu dipublish ke server Pentaho agar dapat diakses. Selain relative mudah penggunaannya, Pentaho Report Designer juga dapat menghasilkan outputan ke banyak format (.pdf, .xls, .html, dll)

5.1 Pengenalan Pentaho Report Designer

Ada beberapa bagian pada antarmuka Pentaho Report Designer

5.1.1 Toolbar item dan penjelasannya

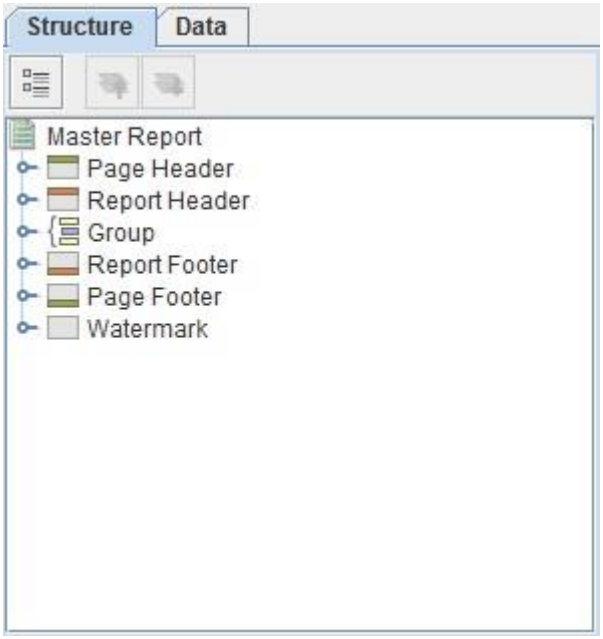
Toolbar item	Description
	Open File
	Save File
	Undo
	Publish
	Redo
	Delete
	Cut
	Copy
	Paste
	Create Report
	Select Font Type
	Select Font Size
	Apply Boldface
	Apply Italics
	Apply Underline
	Select Font Color
	Preview As (PDF, HTML, RTF, XLS, CSV)
	Add Data Source

5.1.2 **Pallete Tool Window**



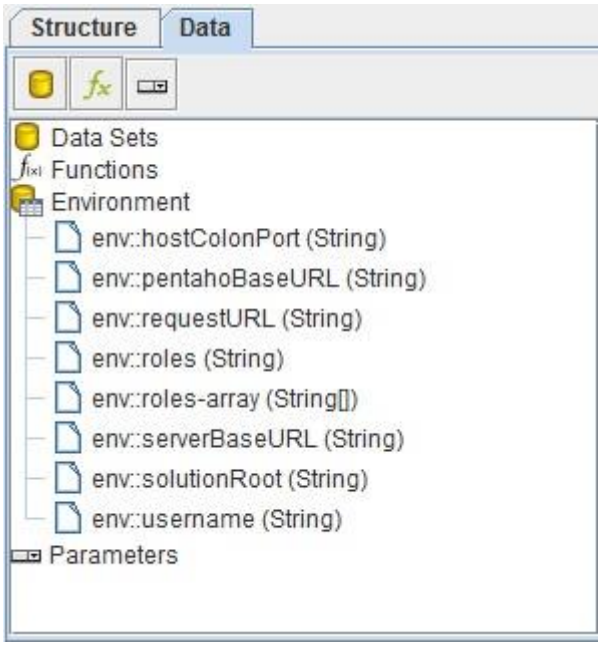
Pallete tool window berisi elemen-elemen yang dapat ditambahkan pada report. Pallete ini berada disebelah kiri dari layar.

5.1.3 **Structure Window**



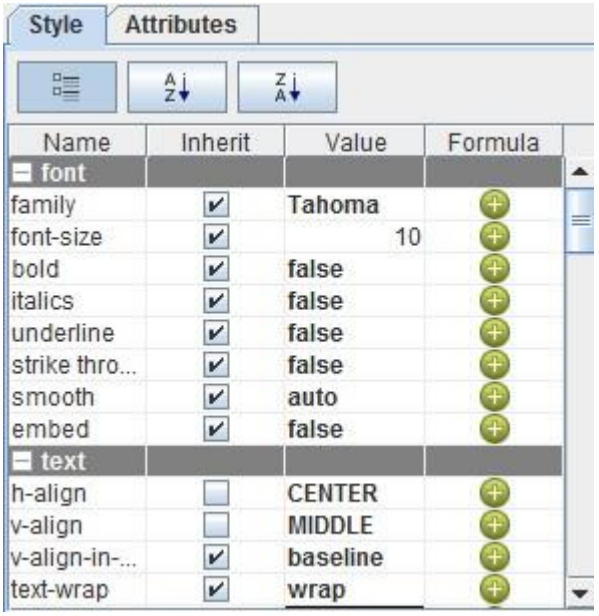
Structure window berfungsi untuk pengaturan struktur pada report. Bagian ini terdapat pada layar sebelah kanan yang bersisian dengan tab ‘Data’.

5.1.4 **Data Window**



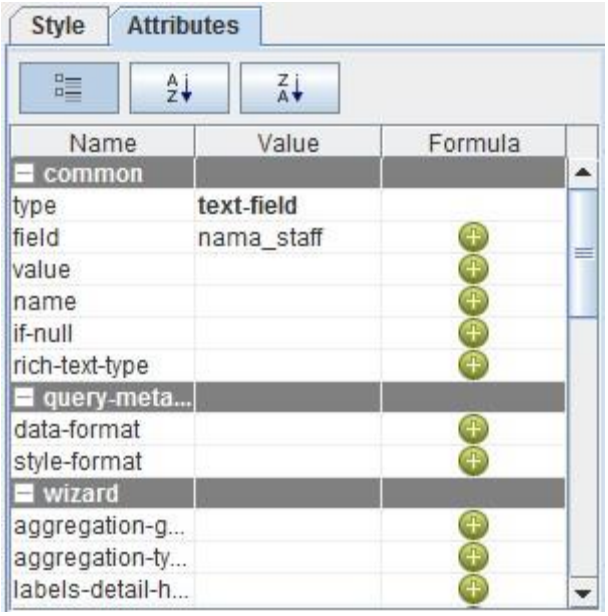
Pada bagian ini untuk pengaturan datasource yang digunakan. Terdapat pada layar sebelah kanan yang bersisian dengan tab
‘Structure’.

5.1.5 **Style Window**



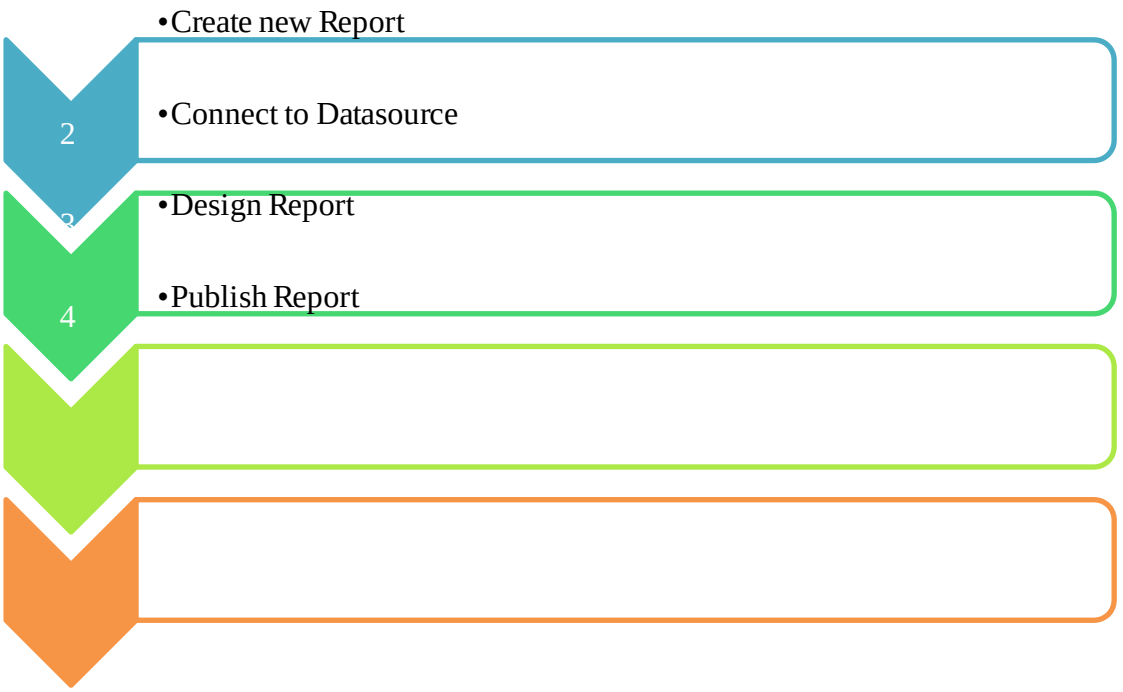
Style window berfungsi untuk pengaturan style (font, text, layout, dan sebagainya). Terdapat pada layar sebelah kanan bawah jika tab 'Structure' dipilih.

5.1.6 **Attributes Window**



Attribute Window berfungsi untuk pengaturan atribut/field yang ada. Terdapat pada layar sebelah kanan bawah jika tab 'Structure' dipilih.

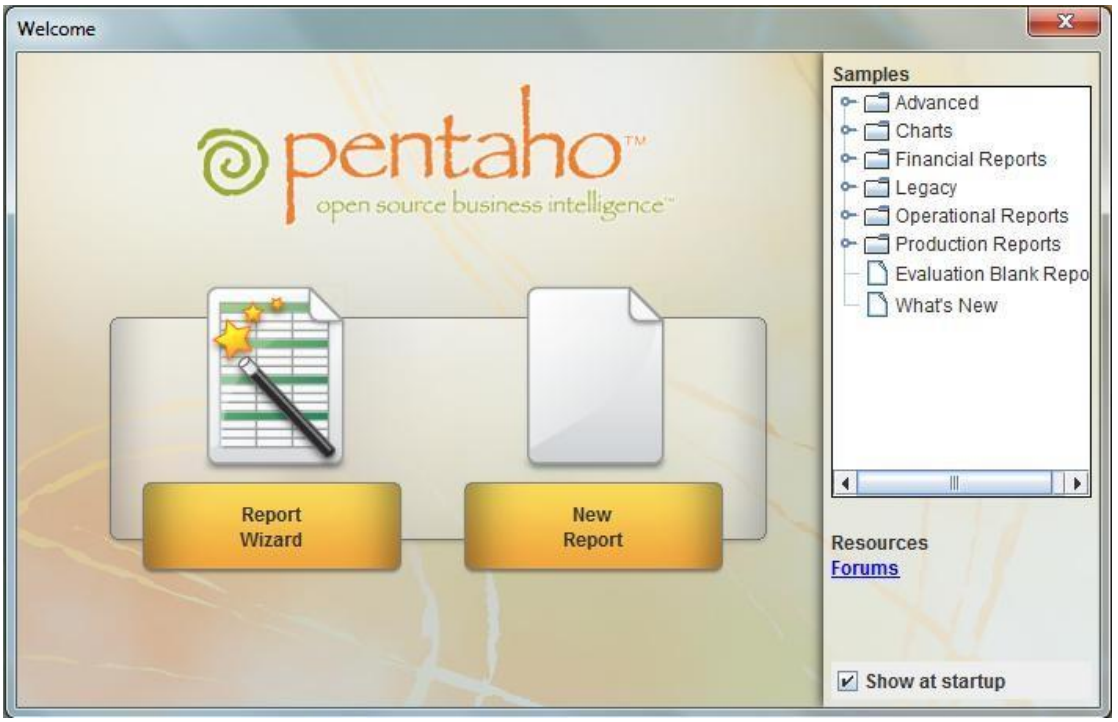
Pada bab ini akan dibahas empat poin tahapan pembuatan report sebagai berikut :



5.2 Creating Report

Dalam pelatihan kali ini kita akan membuat reporting dengan data yang sudah kita olah sebelumnya, yaitu report yang berisi data Sewa Film.

Ada dua cara pembuatan report. Cara pertama sepenuhnya menggunakan designer, dan cara kedua memanfaatkan fasilitas wizard pembangunan report.

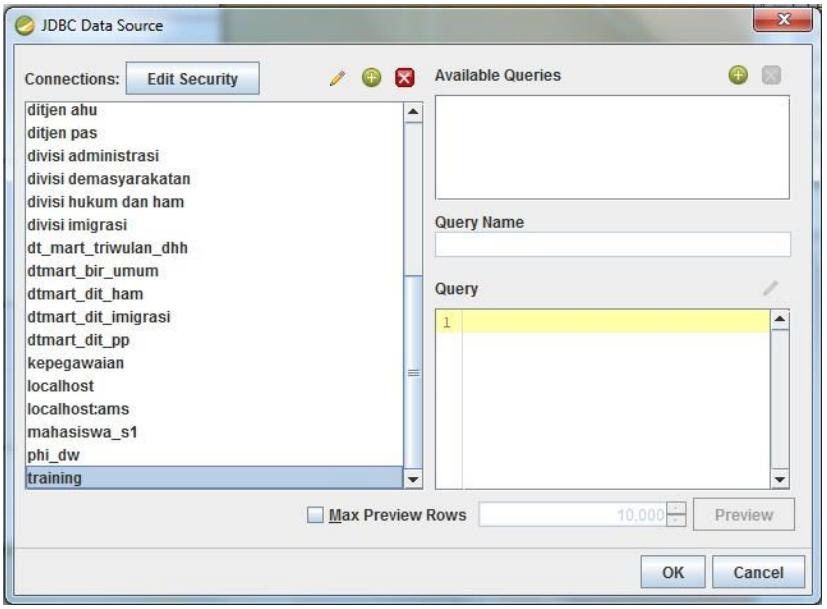


Disain sebuah report yang akan dibandung sebagai berikut :

				tgl cetak
List Penyewaan Film				
Tahun 200x				
No	Tanggal	Judul Film	Biaya Sewa (US $)	Nama Staff

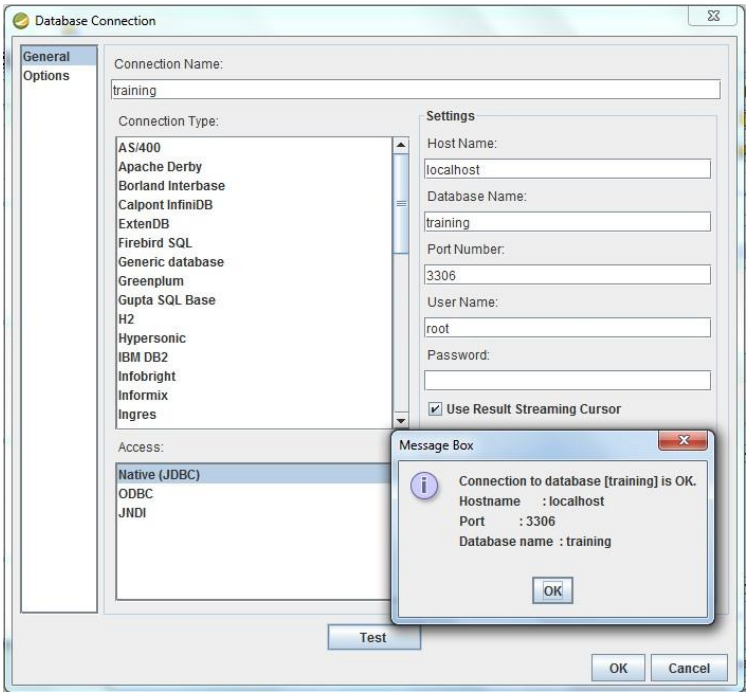
Dengan adanya report ini, diharapkan manager bisa melihat list dari penyewaan yang terjadi.5.2.1 **Cara pertama (tanpa wizard)**

- Klik new report.
- Add datasource .
    - Pilih JDBC.

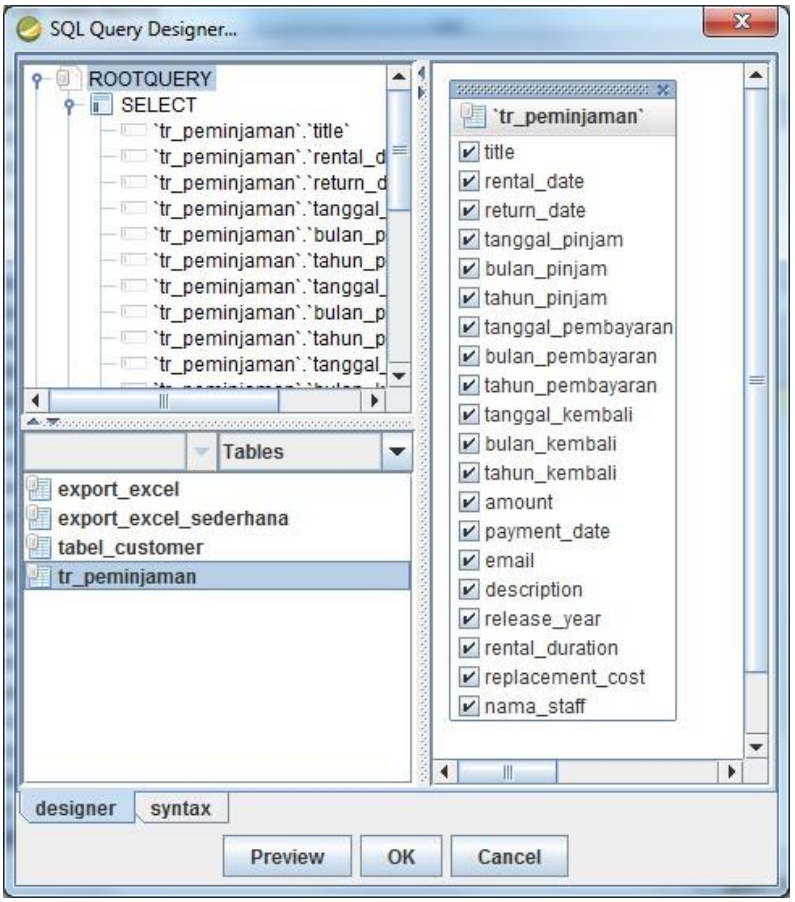


    - Buat koneksi database dengan mengklik  pada bagian connection.

- Isikan konfigurasi koneksi yang akan dibuat lalu dites dengan mengklik tombol ‘Test’.



- Buat query yang sesuai dengan kebutuhan, dan dimasukkan pada panel query. Dengan menggunakan fasilitas query generator, kita dapat membuat query dengan cepat, yaitu dengan mengklik tombol  pada panel query.



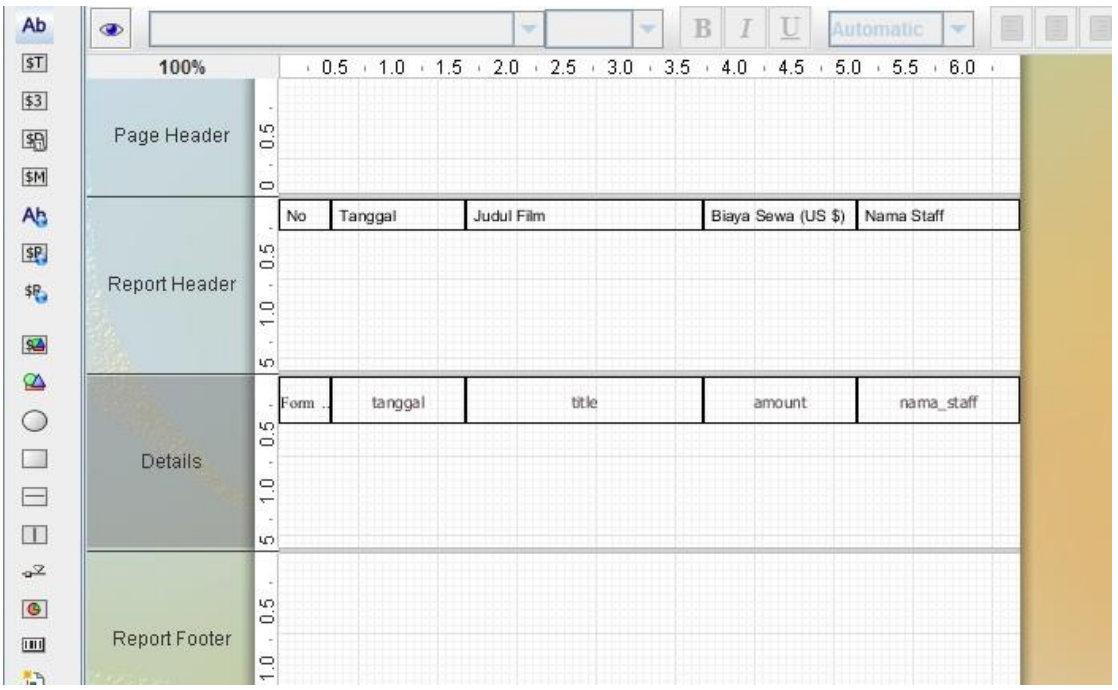
- Tambahkan pada query tersebut sebelum ‘from’

  concat(day(rental_date),'-',monthname(rental_date),'-',year(rental_date)) as tanggal

  tujuannya adalah menambahkan field output yang berbentuk tanggal dengan nama ‘tanggal’

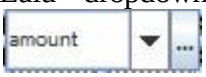
- Setelah query dipreview tidak ada error maka diklik ok.

- Desain Report
  - Buat disain seperti gambar berikut :

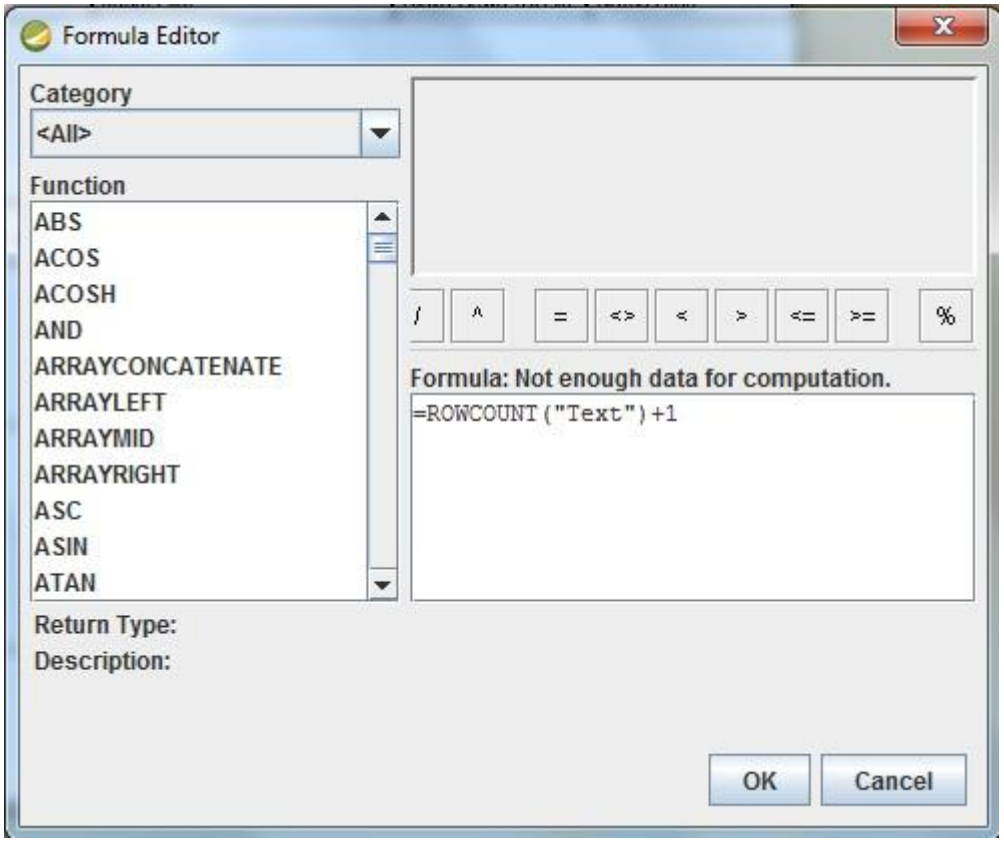


  - Untuk yang berbentuk label, gunakan icon . Untuk yang mengambil dari database (field), gunakan icon .

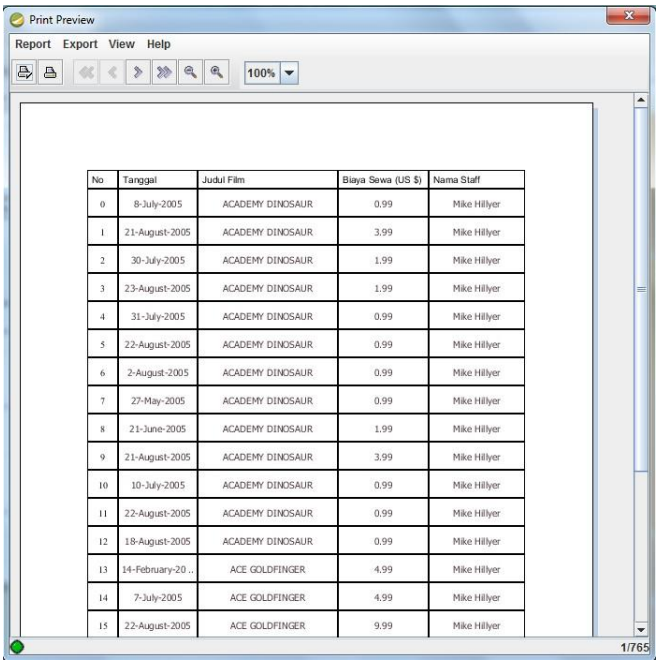
  - Untuk field, pilih field pada database dengan mengklik field tersebut. Lalu dropdown dan pilih field yang diinginkan.



- Untuk penomoran, kita gunakan function. Pembuatan function dengan cara mengklik kanan tulisan function pada tab ‘Data’ lalu pilih ‘Open Formula’ pada kategori ‘Common’. Ketik =ROWCOUNT(‘Text’)+1 seperti gambar berikut :



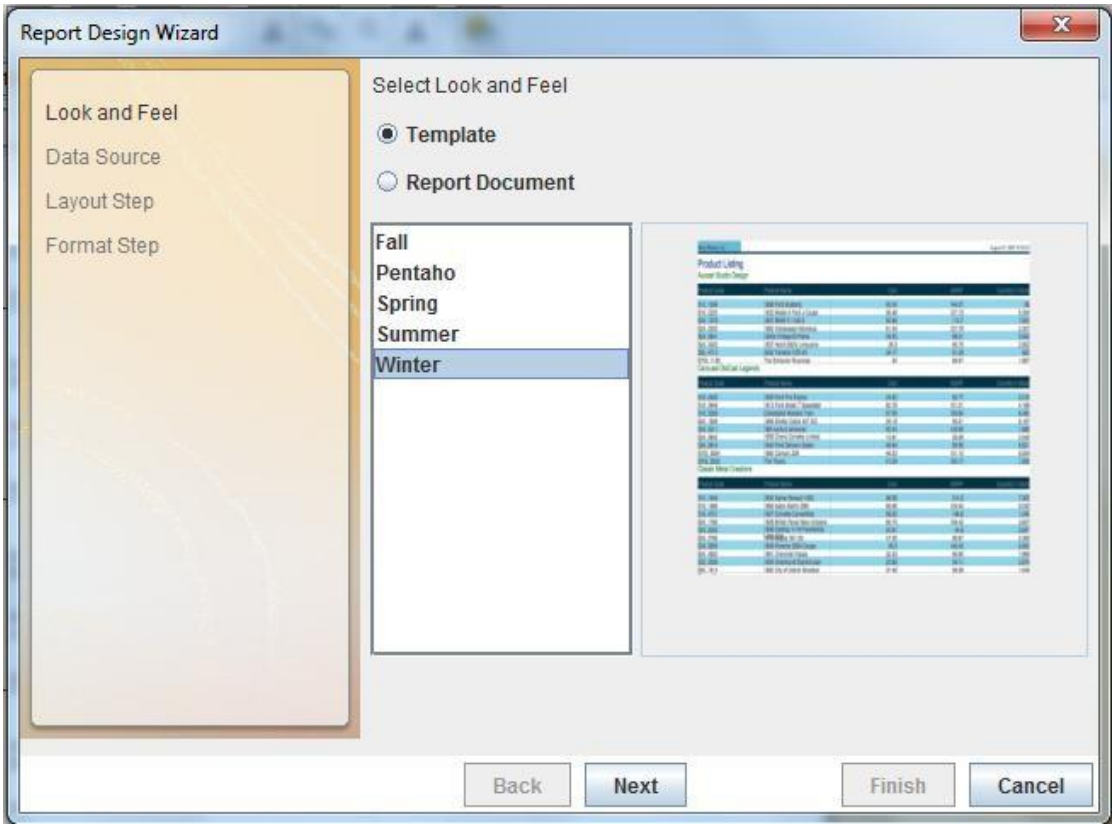
- Preview disain tersebut dengan mengklik tombol . Jika tidak ada kesalahan, maka akan muncul hasil seperti berikut :



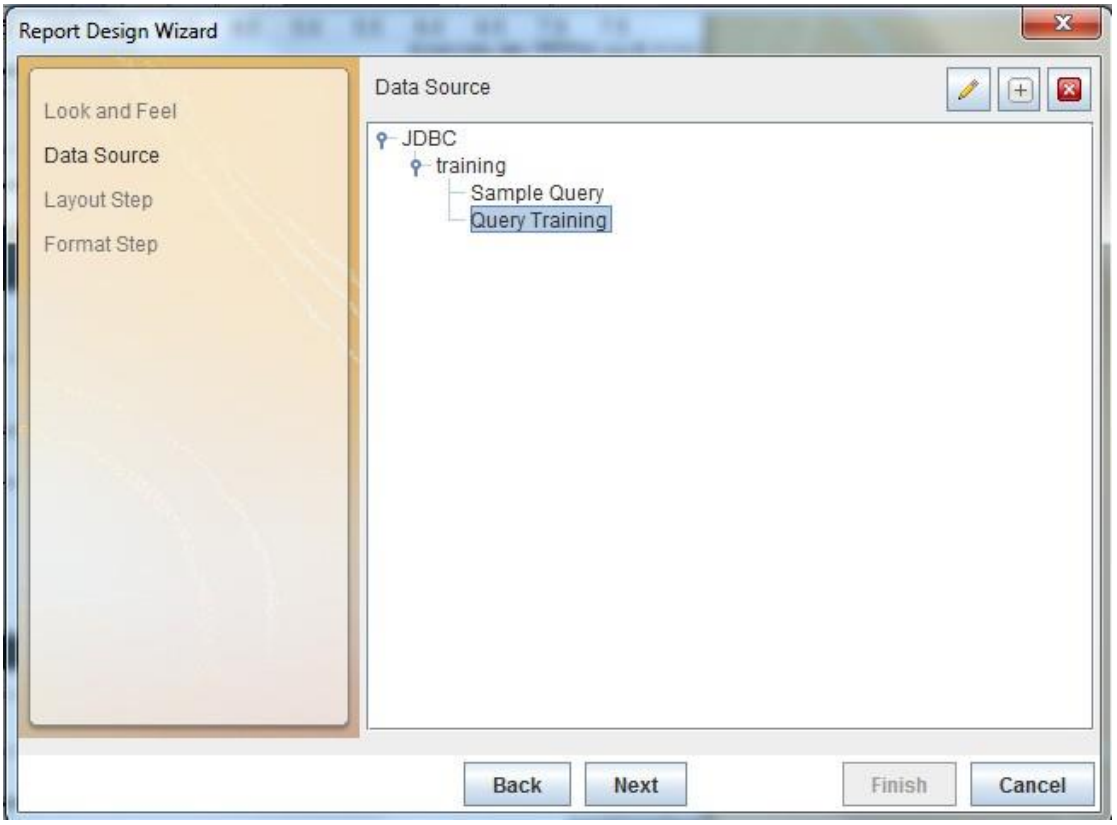
No	Tanggal	Judul Film	Biaya Sewa (US $)	Nama Staff
0	8-July-2005	ACADEMY DINOSAUR	0.99	Mike Hillyer
1	21-August-2005	ACADEMY DINOSAUR	3.99	Mike Hillyer
2	30-July-2005	ACADEMY DINOSAUR	1.99	Mike Hillyer
3	23-August-2005	ACADEMY DINOSAUR	1.99	Mike Hillyer
4	31-July-2005	ACADEMY DINOSAUR	0.99	Mike Hillyer
5	22-August-2005	ACADEMY DINOSAUR	0.99	Mike Hillyer
6	2-August-2005	ACADEMY DINOSAUR	0.99	Mike Hillyer
7	27-May-2005	ACADEMY DINOSAUR	0.99	Mike Hillyer
8	21-June-2005	ACADEMY DINOSAUR	1.99	Mike Hillyer
9	21-August-2005	ACADEMY DINOSAUR	3.99	Mike Hillyer
10	10-July-2005	ACADEMY DINOSAUR	0.99	Mike Hillyer
11	22-August-2005	ACADEMY DINOSAUR	0.99	Mike Hillyer
12	18-August-2005	ACADEMY DINOSAUR	0.99	Mike Hillyer
13	14-February-20 ..	ACE GOLDFINGER	4.99	Mike Hillyer
14	7-July-2005	ACE GOLDFINGER	4.99	Mike Hillyer
15	22-August-2005	ACE GOLDFINGER	9.99	Mike Hillyer

5.2.2 **Cara kedua (dengan wizard)**

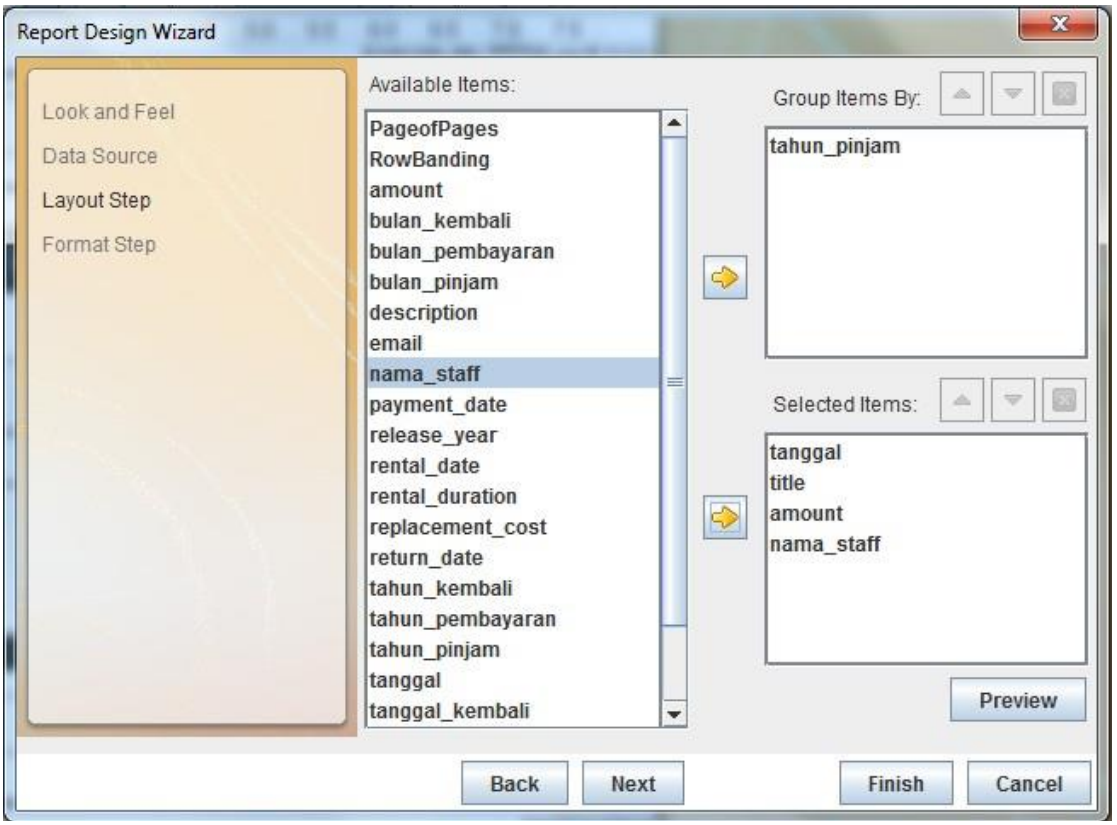
- Klik Report Wizard (saat memulai aplikasi) atau klik menu File, lalu report wizard..
- Pilih template report yang diiinginkan lalu Next.



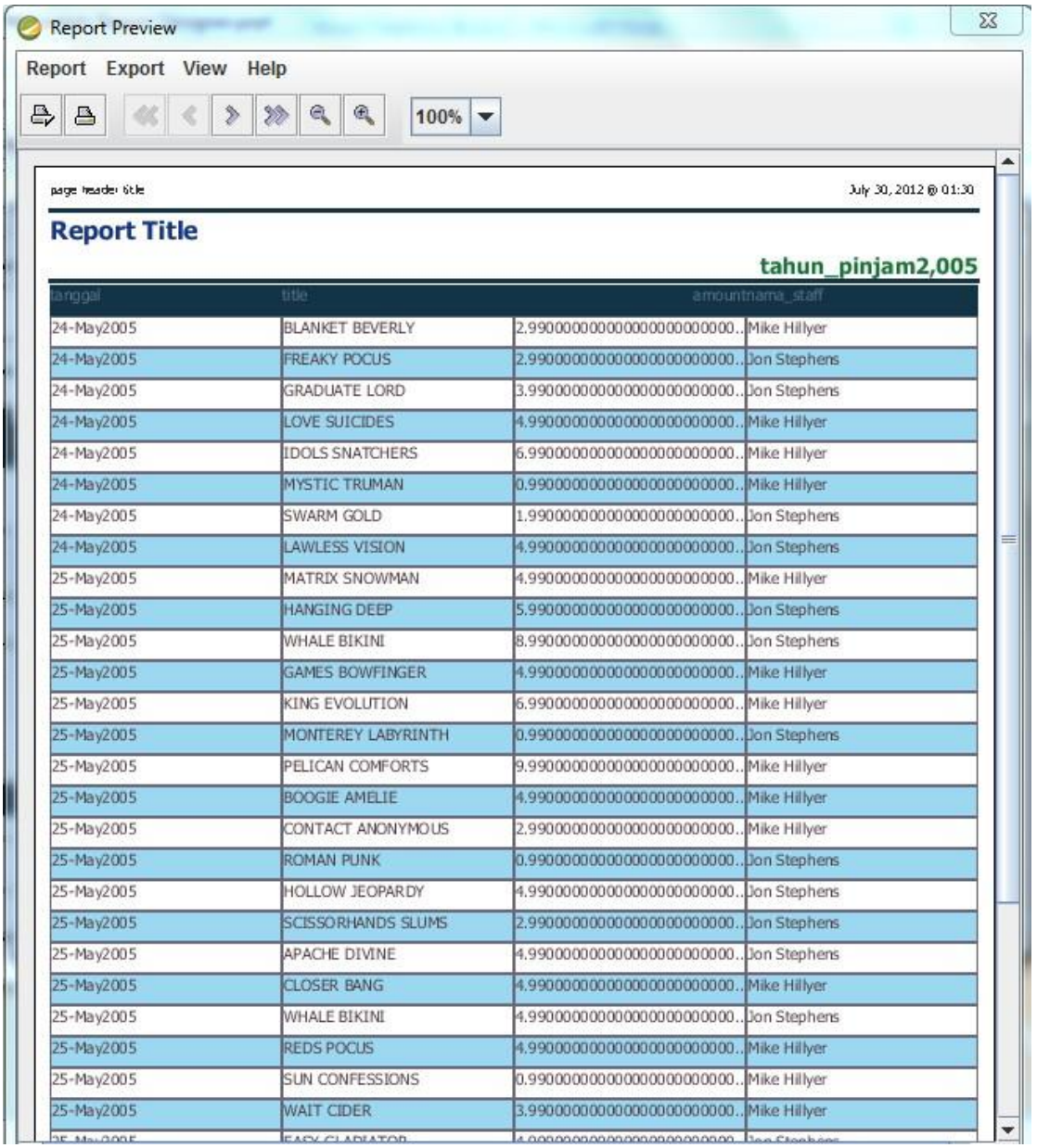
- Definisikan database dengan langkah yang sama seperti cara tanpa wizard.



- Pilih field pada datasource yang ingin kita tampilkan.



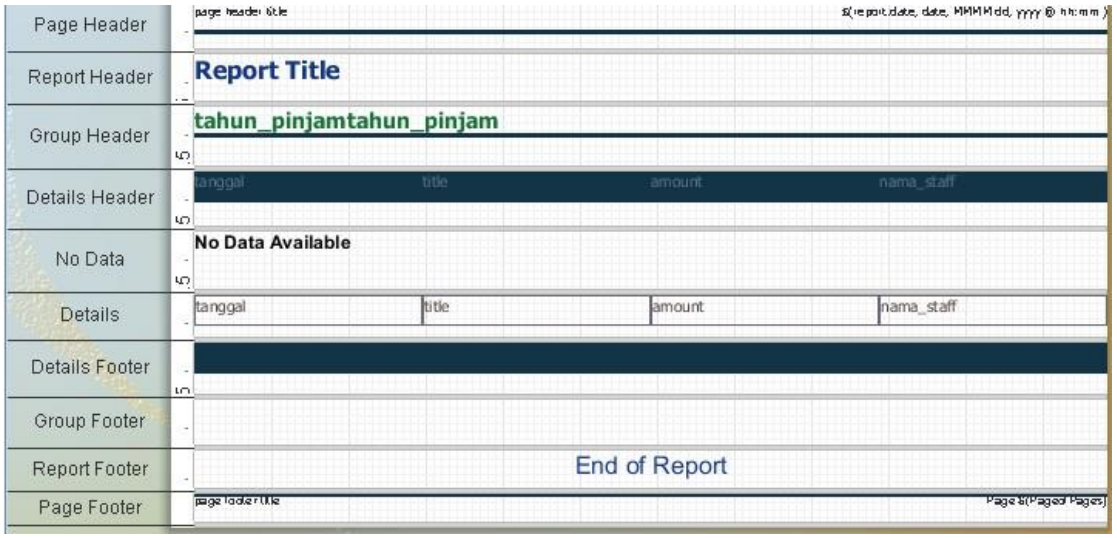
- Preview report tersebut dengan mengklik tombol ‘Preview’



tanggal	title	amountnama_staff	
24-May2005	BLANKET BEVERLY	2.990000000000000000000000..	Mike Hillyer
24-May2005	FREAKY POCUS	2.990000000000000000000000..	Jon Stephens
24-May2005	GRADUATE LORD	3.990000000000000000000000..	Jon Stephens
24-May2005	LOVE SUICIDES	4.990000000000000000000000..	Mike Hillyer
24-May2005	IDOLS SNATCHERS	6.990000000000000000000000..	Mike Hillyer
24-May2005	MYSTIC TRUMAN	0.990000000000000000000000..	Mike Hillyer
24-May2005	SWARM GOLD	1.990000000000000000000000..	Jon Stephens
24-May2005	LAWLESS VISION	4.990000000000000000000000..	Jon Stephens
25-May2005	MATRIX SNOWMAN	4.990000000000000000000000..	Mike Hillyer
25-May2005	HANGING DEEP	5.990000000000000000000000..	Jon Stephens
25-May2005	WHALE BIKINI	8.990000000000000000000000..	Jon Stephens
25-May2005	GAMES BOWFINGER	4.990000000000000000000000..	Mike Hillyer
25-May2005	KING EVOLUTION	6.990000000000000000000000..	Mike Hillyer
25-May2005	MONTEREY LABYRINTH	0.990000000000000000000000..	Jon Stephens
25-May2005	PELICAN COMFORTS	9.990000000000000000000000..	Mike Hillyer
25-May2005	BOOGIE AMELIE	4.990000000000000000000000..	Mike Hillyer
25-May2005	CONTACT ANONYMOUS	2.990000000000000000000000..	Mike Hillyer
25-May2005	ROMAN PUNK	0.990000000000000000000000..	Jon Stephens
25-May2005	HOLLOW JEOPARDY	4.990000000000000000000000..	Jon Stephens
25-May2005	SCISSORHANDS SLUMS	2.990000000000000000000000..	Jon Stephens
25-May2005	APACHE DIVINE	4.990000000000000000000000..	Jon Stephens
25-May2005	CLOSER BANG	4.990000000000000000000000..	Mike Hillyer
25-May2005	WHALE BIKINI	4.990000000000000000000000..	Jon Stephens
25-May2005	REDS POCUS	4.990000000000000000000000..	Mike Hillyer
25-May2005	SUN CONFESSIONS	0.990000000000000000000000..	Mike Hillyer
25-May2005	WAIT CIDER	3.990000000000000000000000..	Mike Hillyer

- Klik Finish.

- Setelah itu akan tampil disain hasil wizard yang telah kita lakukan.



Kita dapat melakukan modifikasi terhadap hasil wizard agar disain lebih sesuai dengan kebutuhan kita.

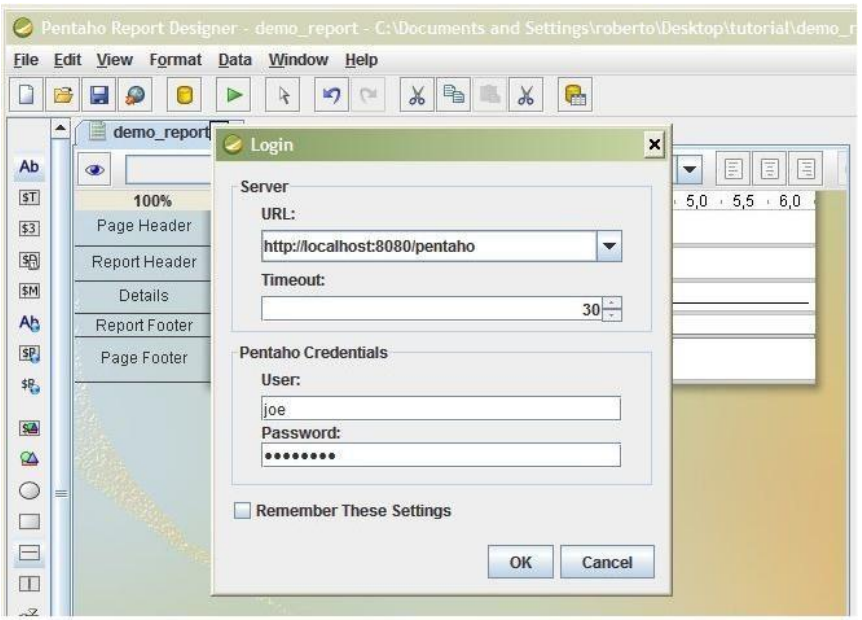
5.3 Publish Report

Setelah design berhasil dipreview, kita publish report tersebut ke server Pentaho yang telah disediakan.

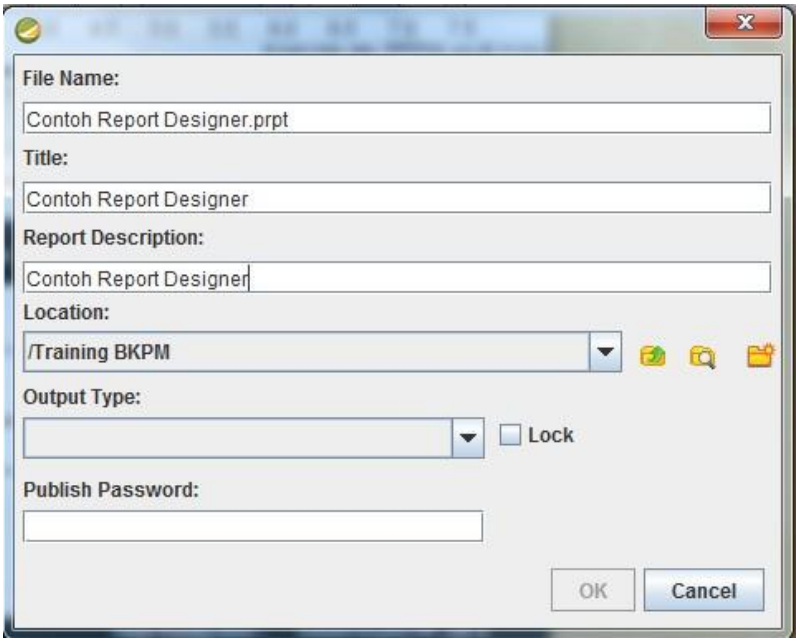
Langkah pertama, buat enable untuk repository access dengan cara mengkonfigurasi file publish_config.xml yang berada di \pentaho-solutions\system\. Isikan dengan password BI Server kita.

```
<publisher-config>
  <publisher-password>your password here</publisher-password>
</publisher-config>
```

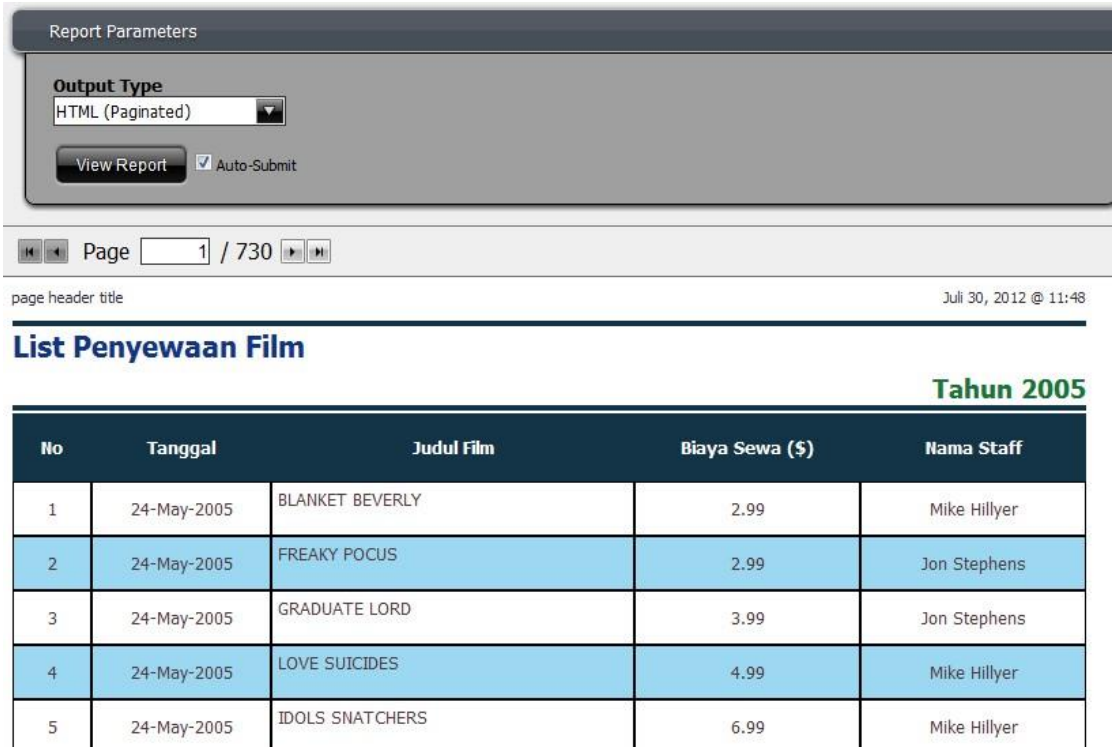
Langkah kedua, klik tombol publish () lalu login agar bisa masuk ke dalam BI Server kita.



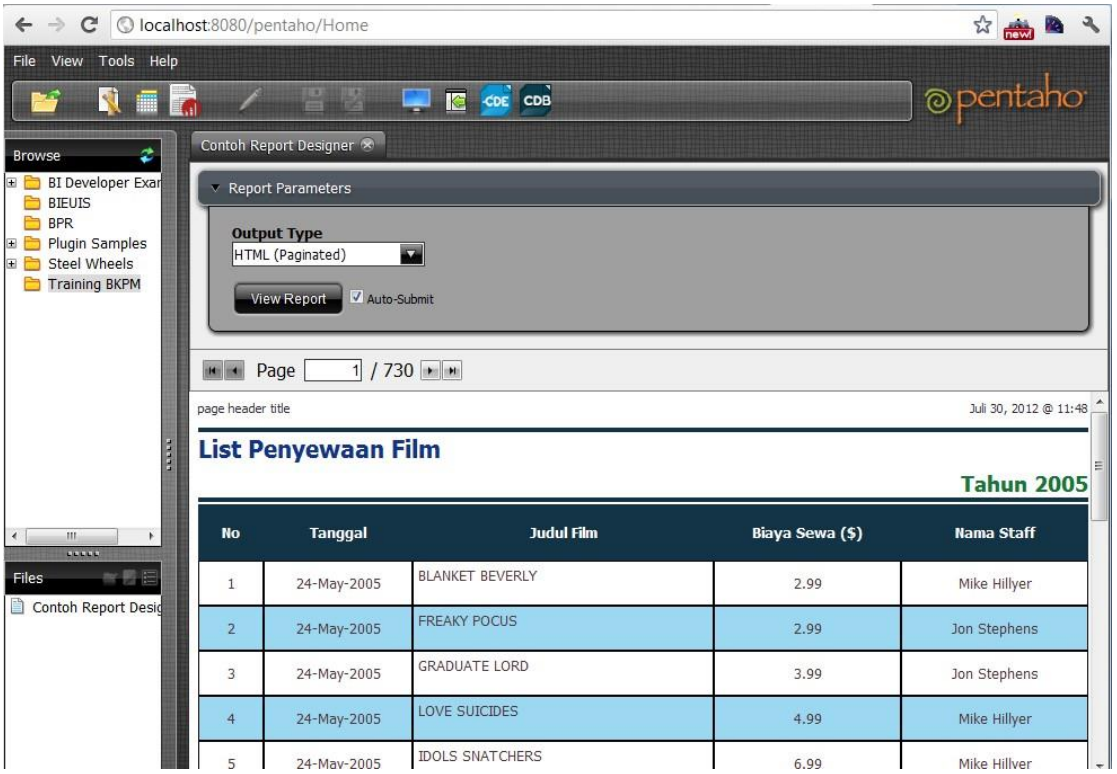
Langkah ketiga, simpan report kita ke folder yang kita inginkan. Kita juga dapat mempublishnya dalam tipe keluaran yang kita inginkan (pdf, excel, html). Bahkan kita dapat me-lock tipe keluaran yang kita inginkan



Berikut adalah contoh report yang telah dipublish



No	Tanggal	Judul Film	Biaya Sewa ($)	Nama Staff
1	24-May-2005	BLANKET BEVERLY	2.99	Mike Hillyer
2	24-May-2005	FREAKY POCUS	2.99	Jon Stephens
3	24-May-2005	GRADUATE LORD	3.99	Jon Stephens
4	24-May-2005	LOVE SUICIDES	4.99	Mike Hillyer
5	24-May-2005	IDOLS SNATCHERS	6.99	Mike Hillyer



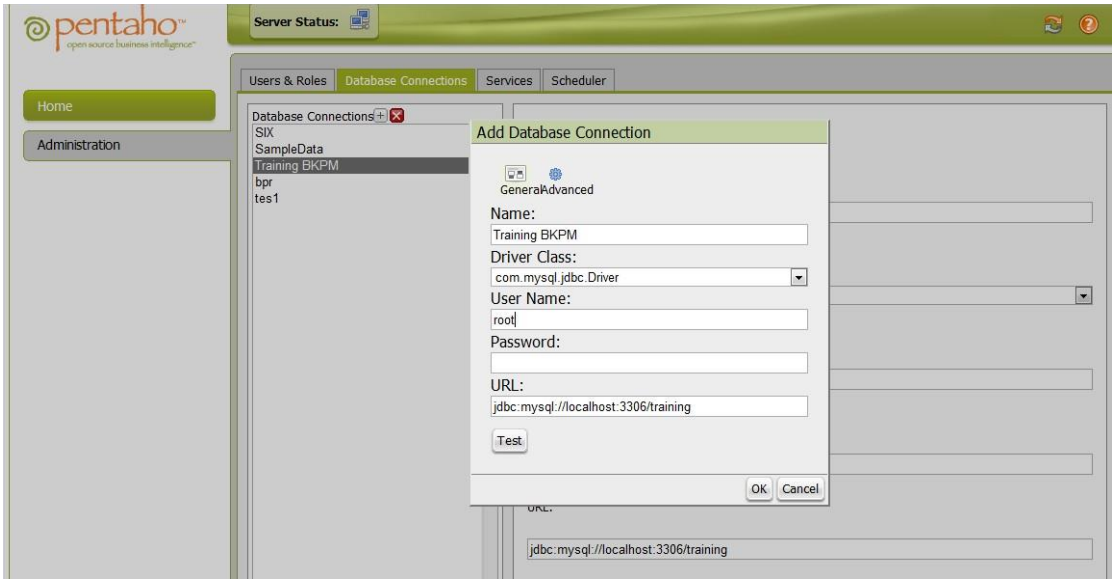
No	Tanggal	Judul Film	Biaya Sewa ($)	Nama Staff
1	24-May-2005	BLANKET BEVERLY	2.99	Mike Hillyer
2	24-May-2005	FREAKY POCUS	2.99	Jon Stephens
3	24-May-2005	GRADUATE LORD	3.99	Jon Stephens
4	24-May-2005	LOVE SUICIDES	4.99	Mike Hillyer
5	24-May-2005	IDOLS SNATCHERS	6.99	Mike Hillyer

Bab 6 Analytic Tools

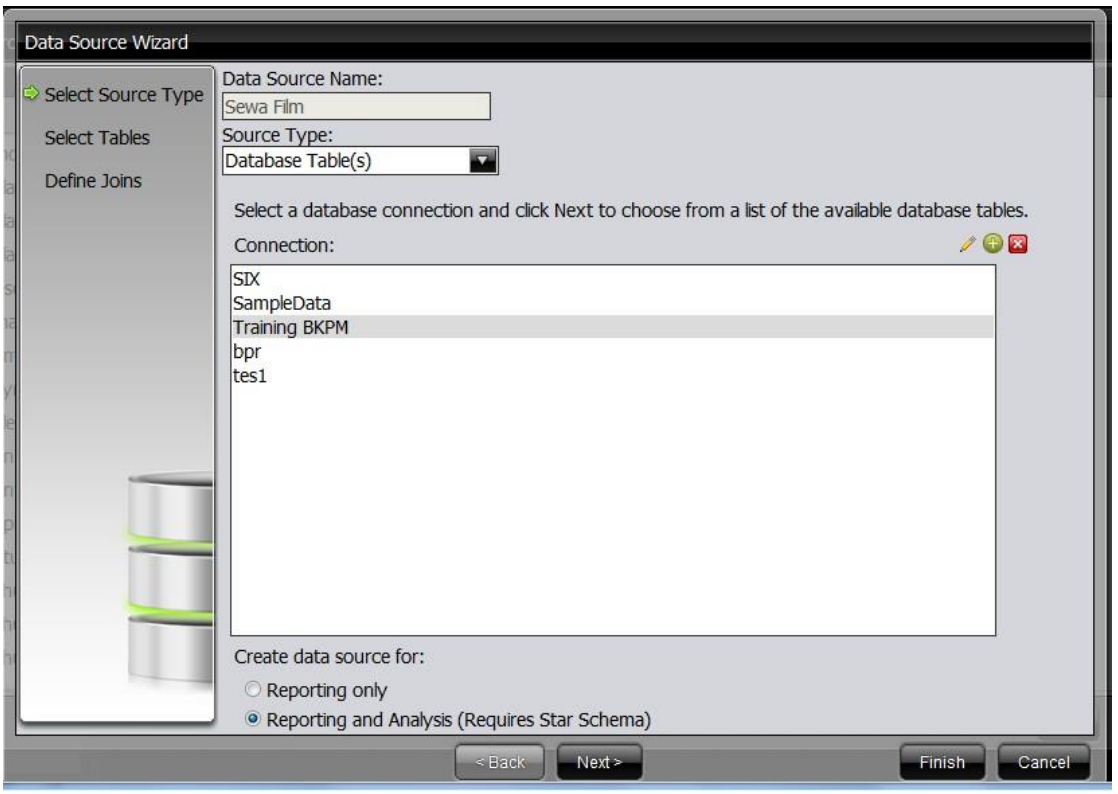
Salah satu core dari Business Intelligence adalah fungsi analisis, yaitu mengolah data agar sesuai dengan kebutuhan. Dengan fungsi ini, kita dapat menentukan field apa saja yang akan dilihat, sebagai row atau columns, dan sebagainya. Di Pentaho BI Server, kita dapat menggunakan tools bawaan Pentaho BI Server Analysis View atau Saiku sebagai plugin.

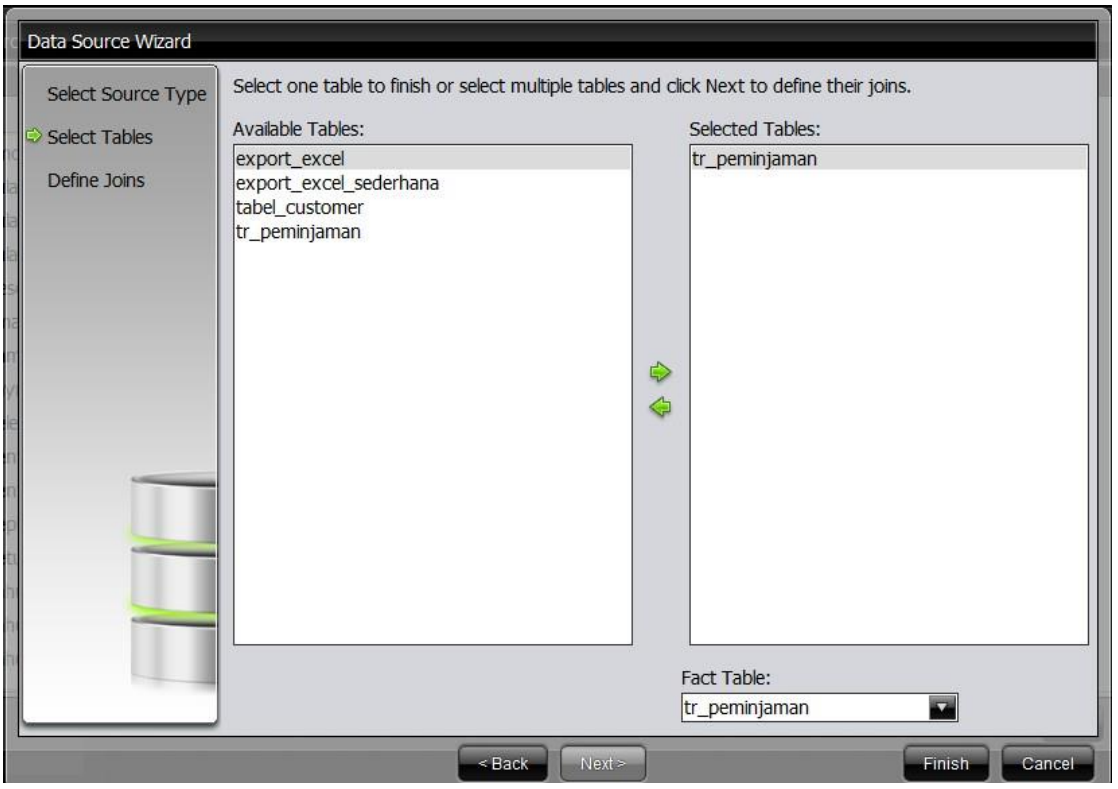
Sebelum melakukan analysis view, harus dipastikan telah dibuat cube/datasource dengan koneksi yang valid. Berikut langkah pembuatan cube :

1. Buat koneksi pada administration console (localhost:8099)

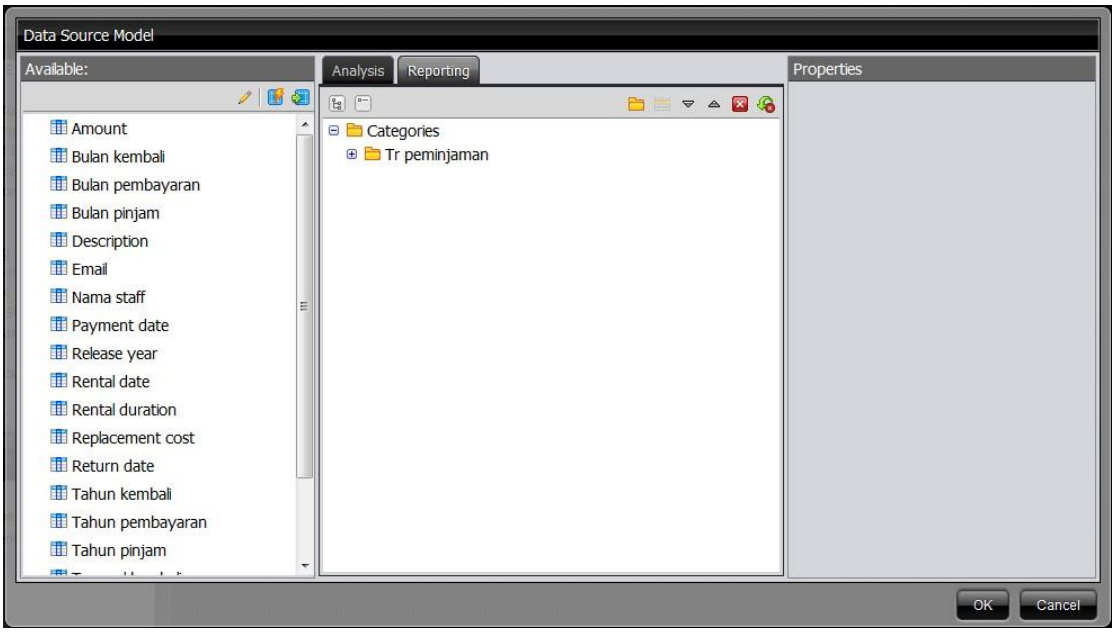


2. Create new datasource pada Pentaho User Console/ BI Server (localhost:8080).





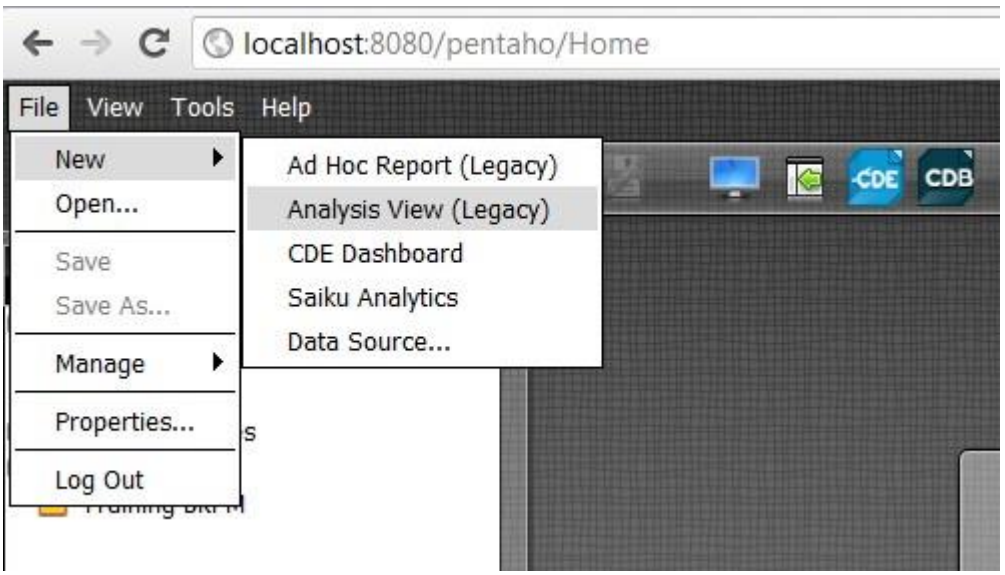
3. Pada Data Source Model, klik OK.



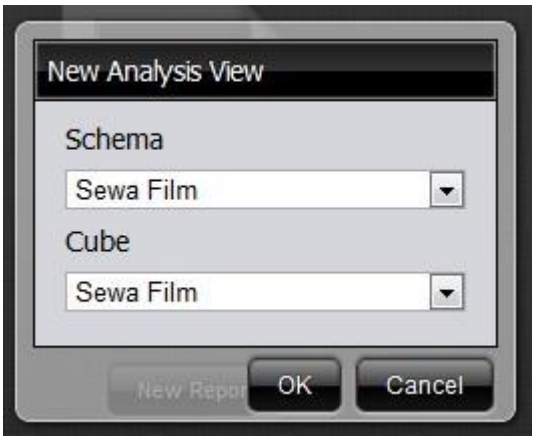
6.1 Pentaho Analysis View

Berikut langkah pemanfaatan analysis view :

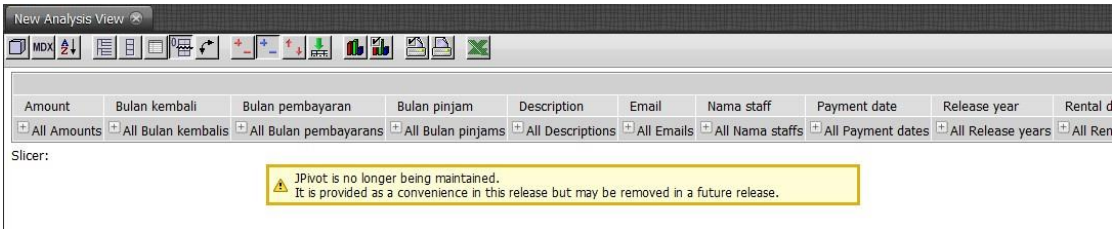
1. Melalui menu file, pilih new file lalu pilih analysis view (legacy).



2. Pilih schema dan cube yang telah kita buat sebelumnya.



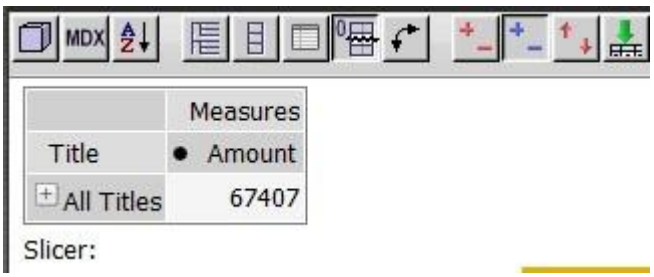
3. Akan muncul default tampilan seperti berikut.



4. Setting cube. Pilih field yang berperan sebagai baris dan kolom. Lalu klik OK



5. Akan muncul tabel analisis.



6. Kita dapat melakukan drill down.

Title	Measures
Title	Amount
ACADEMY DINOSAUR	37
ACE GOLDFINGER	53
ADAPTATION HOLES	38
AFFAIR PREJUDICE	92
AFRICAN EGG	52
AGENT TRUMAN	127
AIRPLANE SIERRA	83
AIRPORT POLLOCK	103
ALABAMA DEVIL	72
ALADDIN CALENDAR	132
ALAMO VIDEOTAPE	36
ALASKA PHANTOM	45
ALI FOREVER	55
ALIEN CENTER	91
ALLEY EVOLUTION	53
ALONE TRIP	63

6.2 Saiku Plugin

Saiku plugin merupakan plugin yang dapat diintegrasi pada BI Server.

Instalasi

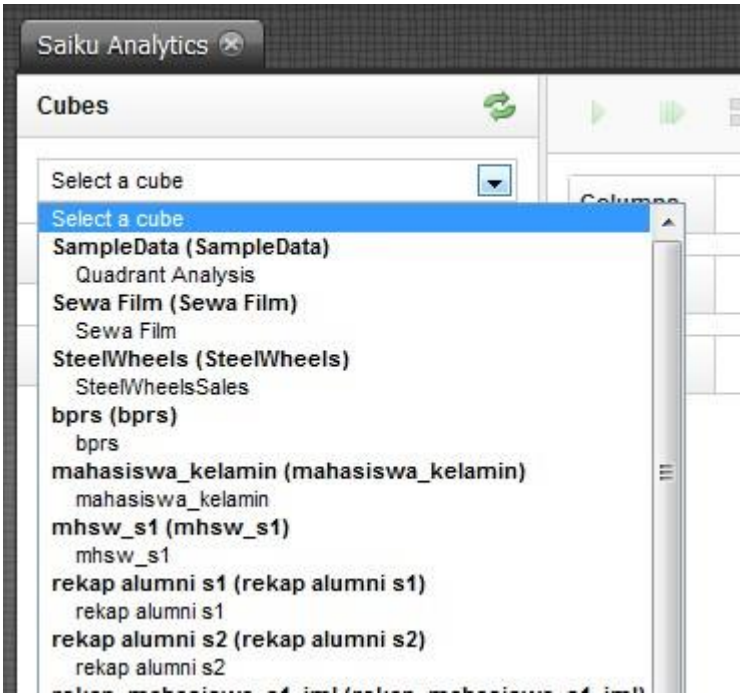
- Untuk mendapatkan file Saiku, kita bisa download dia http://analytical-labs.com/downloads.php.

- Ekstrak file .zip dan dicopy ke lokasi BI Server (biserver-ce\pentaho-solutions\system).

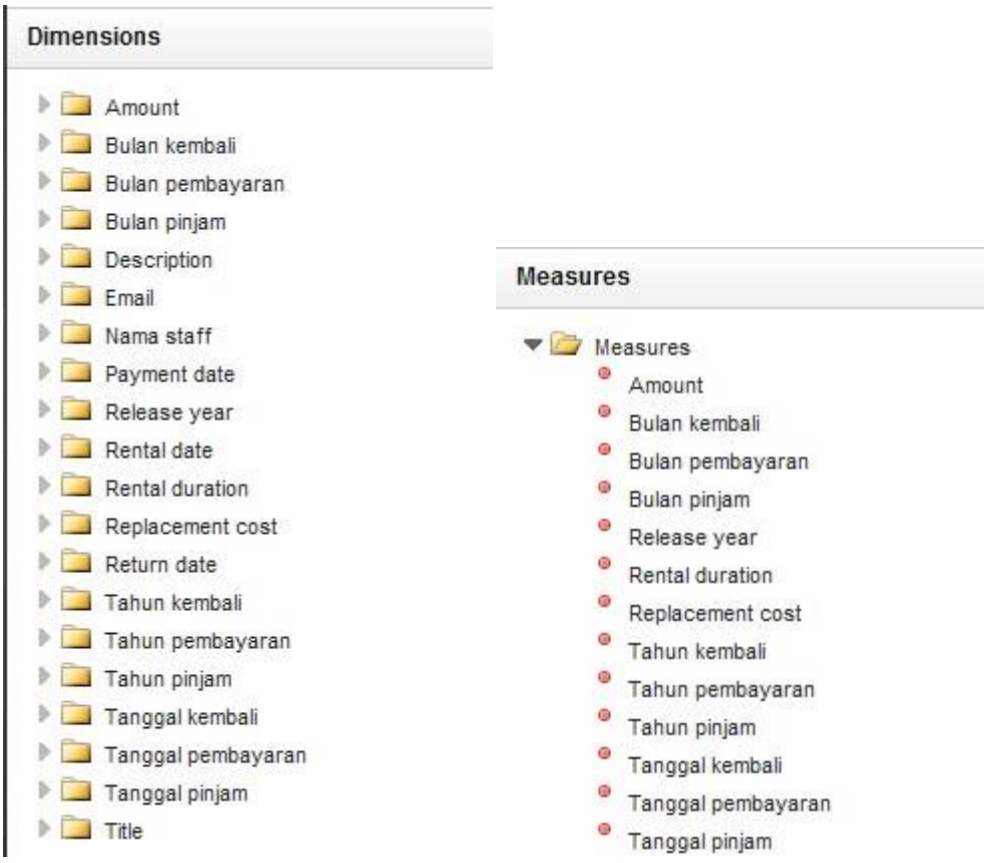


Penggunaan Saiku

- Pilih cube yang akan kita sajikan.



- Pilih dimension dan measure yang akan disajikan.





Tahun pembayaran	Bulan pembayaran	Amount
2005	5	4823
	6	9630
	7	28369
	8	24070
2006	2	514

- Dengan struktur yang sederhana, data dapat disajikan dalam bentuk grafik.

