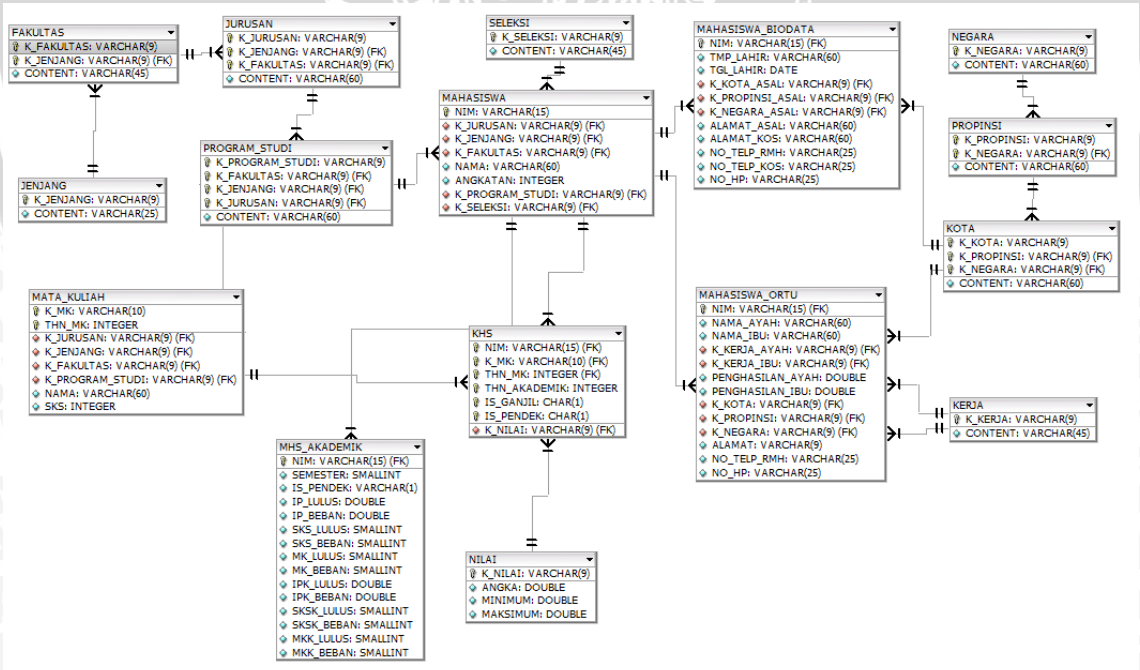
BAB II
TINJAUAN PUSTAKA DAN DASAR TEORI

2.1 PPTI UB (Pengkajian dan Pengembangan Teknologi Informasi Universitas Brawijaya)

PPTI UB merupakan unit khusus yang bergerak dalam riset, pengembangan dan penerapan Teknologi Informasi. PPTI UB dibentuk pada tahun 2001 tetapi telah mengadakan riset sejak tahun 1997. Salah satu rencana pengembangan yang dimiliki PPTI UB adalah seperti pengembangan sarana pendidikan berbasis teknologi informasi dan komunikasi yang terintegrasi [PPT-11]. Dalam penelitian ini sumber data, baik *physical diagram* maupun data OLTP yang digunakan adalah berasal dari PPTI UB, dengan nama dan nim yang disamarkan. Berikut adalah potongan *physical diagram* dari SIAKAD UB (Sistem Informasi Akademik Universitas Brawijaya) yang digunakan sebagai sumber data dalam penelitian ini.



Gambar 2.1 *Physical Diagram Source*

(Sumber : PPTI-UB)

Dari tabel – tabel yang terdapat pada *physical diagram* diatas, hanya beberapa yang akan digunakan dalam penelitian ini, tabel – tabel tersebut adalah sebagai berikut :

1. TABEL FAKULTAS
2. TABEL JURUSAN
3. TABEL PROGRAM STUDI
4. TABEL JENJANG
5. TABEL MAHASISWA
6. TABEL SELEKSI
7. TABEL MAHASISWA_BIODATA
8. TABEL KOTA
9. TABEL PROPINSI
10. TABEL NEGARA
11. TABEL MHS_AKADEMIK

2.2 DATA

Data merupakan deskripsi dari suatu benda, peristiwa, aktivitas dan transaksi yang direkam, dikelompokkan, dan disimpan namun belum diorganisasikan sehingga belum dapat menyampaikan arti tertentu. [TRL-10]. Data adalah kumpulan fakta atau bahan observasi mentah berupa fenomena atau transaksi bisnis [OBJ-08]. Data dapat diartikan juga sebagai kumpulan fakta, konsep, atau instruksi yang berguna untuk komunikasi, perbaikan dan diproses otomatis agar dapat merepresentasikan informasi sehingga dapat dimengerti dan dipahami manusia [IWH-05]. Data juga dapat diartikan sebagai catatan, fakta – fakta mentah tentang manusia, tempat kejadian dan benda yang belum diolah sehingga belum dapat dipahami dan difungsikan secara maksimal untuk pemecahan masalah [BNS-13].

Berdasarkan teori-teori yang ada di atas, dapat disimpulkan bahwa data merupakan catatan, deskripsi, fakta mentah, bahan observasi tentang sesuatu yang belum diolah sehingga belum dapat dipahami dan belum dapat digunakan secara matang untuk pemecahan masalah, namun bila diproses akan dapat merepresentasikan sesuatu sehingga mudah dipahami.

2.3 INFORMASI

Data dapat memiliki suatu nilai yang disebut informasi apabila telah diolah. Informasi dapat diartikan sebagai sekumpulan data yang sudah diolah dalam suatu proses sehingga yang awalnya tidak memiliki nilai menjadi memiliki nilai yang bermanfaat bagi penggunanya [TRK-06]. Informasi merupakan kumpulan data yang ditempatkan dalam suatu konteks yang memiliki suatu arti dan dapat berguna untuk pengguna akhir [OBJ-08]. Informasi berasal dari data yang telah diolah yang berguna untuk menyelesaikan masalah juga mengambil keputusan [BNS-13].

Berdasarkan teori-teori yang ada di atas, dapat disimpulkan bahwa informasi adalah data yang telah diolah oleh suatu sistem tertentu sehingga memiliki nilai dan dapat digunakan untuk menyelesaikan masalah tertentu atau mengambil keputusan bagi pengguna akhirnya.

2.4 DATABASE

Database bisa diartikan sebagai kumpulan data yang memiliki relasi secara logis dan dapat menghasilkan informasi setelah menjalani proses dalam sistem [BNS-13]. D*atabase* adalah kumpulan relasi (hubungan) data yang logis yang didesain untuk memenuhi kebutuhan bisnis perusahaan [CNB-10]. *Database* juga dapat kita artikan kumpulan data yang berelasi dan disimpan berdasarkan skema yang ada (biasanya minim redudansi) [IWH-05].

Berdasarkan teori-teori yang ada di atas, dapat disimpulkan bahwa *database* merupakan kumpulan data – data yang saling berelasi atau berhubungan secara logis yang dapat digunakan untuk menampilkan informasi setelah melalui berbagai proses di dalam sistem yang dapat memenuhi kebutuhan bisnis suatu organisasi atau perusahaan dan disimpan berdasarkan skema yang ada.

2.5 ALJABAR RELASIONAL

Aljabar relasional merupakan satu dari dua bahasa *query* formal yang berhubungan dengan model relasional. *Query* pada aljabar relasional menggunakan sekumpulan *operator*. Setiap *query relasional* mendeskripsikan

step – step dari suatu *procedure* untuk menghitung jawaban yang diinginkan. Berdasarkan permintaan dari operator yang digunakan dalam *query* [RRG-00].

Aljabar Relasional merupakan kumpulan dari operasi terhadap relasi yang mana setiap operasi dapat menggunakan satu atau lebih relasi agar dapat menghasilkan relasi yang baru juga termasuk pada kategori *procedural* serta juga menyediakan beberapa operator yang dapat digunakan untuk memanipulasi data. Berbagai operasi pada aljabar relasional [RRG-00] :

- *Selection* dan *Projection*

  Aljabar relasional mencakup *operator* untuk *select rows* dari suatu relasi (σ) dan untuk *project* kolom (π). Operator tersebut mengijinkan untuk memanipulasi data pada *single relation*.

- *Set Operation*

  Beberapa operasi standar juga dapat digunakan dalam aljabar relasional, yaitu adalah sebagai berikut :

  - Union (∪)

    Mengembalikan *instance* relasi yang berisi semua tuples yang ada pada relasi pertama atau relasi kedua. Syarat yang harus dipenuhi agar dapat menjalankan *union* adalah yang pertama memiliki jumlah *fields* yang sama dan masalah kecocokan *fields*, dari kiri ke kanan harus memiliki *domain* yang sama.

  - *Intersection* (∩)

    Mengembalikan *instance* relasi yang berisi semua *tuples* yang ada pada relasi pertama dan relasi kedua.

  - *Set-difference* (-)

    Mengembalikan *instance* relasi yang berisi semua *tuples* yang ada pada relasi pertama tapi yang tidak ada di relasi kedua.

  - *Cross product* (x)

    Mengembalikan instance relasi yang mana *schema*nya berisi semua *fields* di relasi pertama diikuti dengan semua fields yang ada pada relasi kedua.

- *Renaming* (ρ)
  Operasi untuk melakukan perubahan nama.
- *Joins*
  Operator yang dapat digunakan untuk mengkombinasikan informasi dari dua atu lebih relasi. Beberapa *varian* dari *join* adalah sebagai berikut:
  - Condition Joins
    Join Condition sama artinya dengan *selection condition* pada suatu bentuk (*cross product* dengan kondisi).
  - *Equijoin*
    Join yang sifatnya menggabungkan 2 relasi yang memiliki *field* yang sama maka hanya salah satu yang ditampilkan (relasi pertama).
  - *Natural Join*
    Sifat yang ada pada *natural join* adalah menyamakan kedua relasi bila memiliki *field* yang saling berhubungan.
- *Division* (÷)
  Menampilkan relasi yang pertama dimana relasi pertama tersebut memiliki *field* yang mengandung semua yang ada di relasi kedua.

2.6 OLTP (*ONLINE TRANSACTION PROCESSING*)

OLTP merupakan bagian yang penting dalam kegiatan operasional dan transaksional keseharian perusahaan. OLTP adalah data operasional dari aktivitas dan sistem yang saling berelasi dengan memasukkan data yang benar dan sesuai kedalam *database* dan kebanyakan merujuk pada *relational database* [IWH-05]. OLTP dapat juga diartikan sebuah sistem yang dirancang atau dibuat agar dapat memaksimalkan kapasitas pemrosesan pada transaksi yang berlangsung [CNB-10].

Berdasarkan teori-teori yang ada di atas, maka dapat disimpulkan bahwa OLTP adalah sebuah sistem yang dirancang atau dibuat untuk memasukkan data yang benar dan sesuai ke dalam *database* dan dirancang atau dibuat untuk

memaksimalkan kapasitas pemrosesan pada transaksi serta kebanyakan merujuk pada *relational* database.

2.7 OLAP (*ONLINE ANALYTICAL PROCESSING*)

Menurut *Kimball,* OLAP adalah suatu istilah dari suatu teknologi dengan gambaran multidimensional dimana dapat menyediakan akses yang cepat dengan informasi strategis yang berguna untuk keperluan analisis lanjutan [KRR-10]. Bagi pengguna, OLAP memungkinkan untuk bisa mendapatkan pemahaman dan pengertian yang lebih dalam dan berbagai pengetahuan mengenai aspek data dengan cepat dan konsisten serta interaktif dalam berbagai sudut pandang terhadap data [BNS-13]. Sementara sistem OLAP dapat menjawab pertanyaan yang membedakan mereka dari *query tools* yang umum, artinya OLAP dapat menjawab *business question* yang diajukan. OLAP dapat memberikan tampilan yang ringkas dan singkat, namun terdapat cara baru pula untuk melihat data dalam *Bussiness Inteligence*. OLAP *cube* merupakan salah satu sistem OLAP yang paling unggul [BNS-13]. Dalam OLAP *cube,* penyimpanan data dirancang secara spesial agar dapat menghadapi data secara spesifik dan berbentuk ringkasan multidimensional yang dapat diwujudkan dengan struktur dua dimensi (gambar 2.1), atau tiga dimensi (gambar 2.2). [BNS-13]

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)

City

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

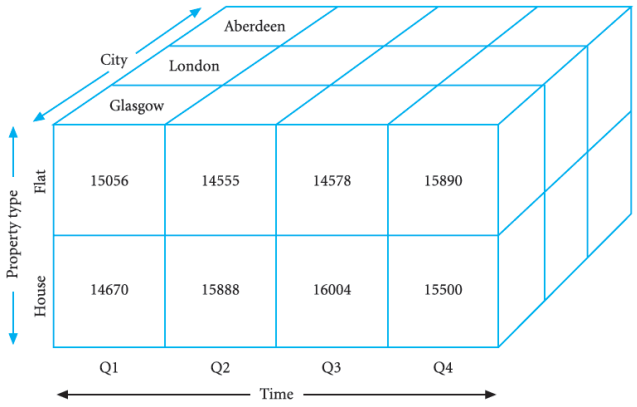
(b)

Gambar 2.2 OLAP *Cube* 2 dimensi

(Sumber : [CNB-10])

Property Type	City	Time	Total Revenue
Flat	Glasgow	Q1	15056
House	Glasgow	Q1	14670
Flat	Glasgow	Q2	14555
House	Glasgow	Q2	15888
Flat	Glasgow	Q3	14578
House	Glasgow	Q3	16004
Flat	Glasgow	Q4	15890
House	Glasgow	Q4	15500
Flat	London	Q1	19678
House	London	Q1	23877
Flat	London	Q2	19567
House	London	Q2	28677
.....	.....	.....	.....
.....	.....	.....	.....

(c)



(d)

Gambar 2.3 OLAP *Cube* 3 dimensi

(Sumber : [CNB-10])

Adapun beberapa karakteristik dari OLAP adalah sebagai berikut **[OPR-12]**:

- Pengguna dapat melakukan analisis *query* yang interaktif dan kompleks.
- Dapat membuat para pelaku bisnis memiliki pandangan *logical* dan multidimensional terhadap data di dalam *data warehouse.*
- Dapat membuat *user* melakukan *drill down* agar mendapatkan rincian yang lebih jelas atau *roll up* untuk agregasi *metric* dalam satu dimensi atau multidimensi bisnis.
- Terdapat kemampuan untuk dapat menampilkan kalkulasi yang rumit dan perbandingan.

2.8 DATA WAREHOUSE

Data warehouse dapat didefinisikan sebagai sekumpulan data yang memiliki karakteristik *subject-oriented, Data Integrated, time variant* dan *nonvolatile* sebagai implementasi fisik dari *data model* untuk mengambil keputusan dan menyimpan informasi untuk keputusan strategis dalam kebutuhan perusahaan [RGS-10]. *Data warehouse* memiliki beberapa teknologi yaitu pembersihan data, integrasi data, dan OLAP untuk teknik analisis yang berfungsi untuk menyimpulkan, meringkas dan agregasi

seperti kemampuan untuk melihat informasi dengan banyak sudut pandang [RGS-10]. *Data Warehouse* juga dapat diartikan sebagai integrasi berbagai informasi dari beberapa operasional *database* dan seringkali digunakan untuk *support Online Analytical Processing* (OLAP) [CMY-01]. Jika dapat disimpulkan dari teori – teori diatas maka *data warehouse* merupakan integrasi informasi dari beberapa operasional database yang bersifat *subject-oriented*, terintegrasi, *time variant* dan *non-volatile*, dimana nantinya dapat digunakan untuk memandang informasi dari berbagai sudut pandang sehingga dapat digunakan untuk membantu mengambil sebuah keputusan dan seringkali digunakan untuk *support* OLAP.`

2.8.1 KARAKTERISTIK *DATA WAREHOUSE*

Data warehouse memiliki perbedaan yang cukup terlihat apabila dibandingkan dengan OLTP (*Online Transaction Processing*). Beberapa perbedaan dapat dilihat di tabel berikut ini.

Tabel 2.1 Perbedaan OLTP dengan *Data Warehouse*

(Sumber : [CNB-10])

Karakteristik	OLTP	*Data Warehouse*
Main Purpose	Dapat mendukung proses operasional.	Dapat mendukung proses Analisis
Data Age	Menangani Data Sekarang (Saat ini)	Menangani Data yang Historikal
Data Granularity	Menyimpan Data yang detail	Menyimpan Ringkasan sederhana dan lanjutan, Data yang detail

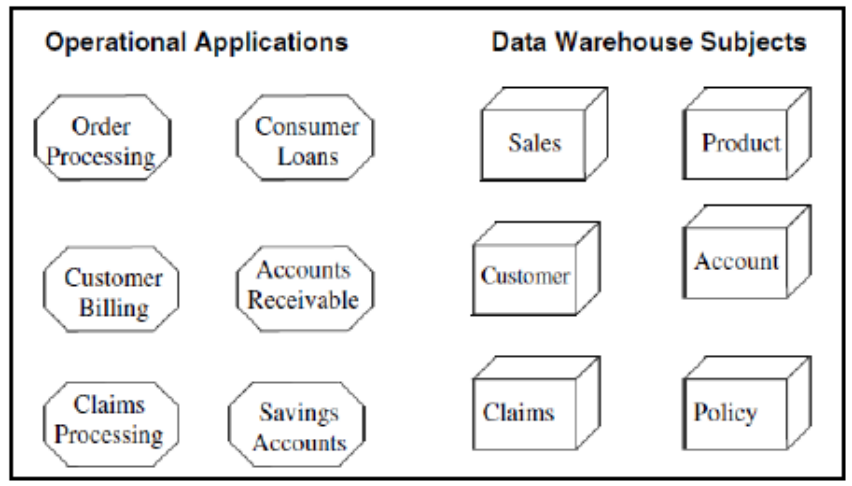
Data Processing	Masih terdapat Pola pengubahan dan penambahan, serta pada *query* pengurangan data masih dapat diprediksi	Pola *query* data lsulit diprediksi
Reporting	Masih dapat diprediksi, tidak multidimensi (1 dimensi)	Sulit diprediksi, multidimensi (lebih dari 1 dimensi)
Orientasi	Sistem ini berorientasi aplikasi	Sistem ini berorientasi subjek
Users	Mendukung para pengguna operasional	Mendukung manajer mengambil keputusan
Sifat Data	Data bersifat Dinamis (dynamic)	Data bersifat statis (static)

Beberapa penjelasan karakteristik *Data Warehouse* berikut ini :

1. *Subject Oriented*

Data warehouse memiliki karakteristik *Subject Oriented*, maksudnya adalah *data warehouse* disusun atau didesain untuk menganalisa data berdasarkan subjek – subjek utama suatu *database* dari sebuah perusahaan, dimana pada setiap fisikal subjek diimplementasikan sebagai kumpulan dari tabel – tabel pada *database* yang berhubungan dalam *data warehouse* dan tidak berdasarkan pada proses atau fungsi aplikasi tertentu. [IWH-05]. *Data Warehouse* diorganisasikan dalam lingkup subjek dan berfokus ke arah

permodelan dan analisis data bagi pihak – pihak pembuat atau pengambil keputusan [PNP-01]. Sehingga berdasarkan teori – teori diatas, dapat disimpulkan bahwa pada karakteristik ini, *data warehouse* didesain atau disusun untuk menganalisa data berdasarkan subjek – subjek utama pada *database* (organisasi) bukan pada fungsi aplikasi tertentu sehingga dapat berfokus ke arah permodelan dan analisis data. Pada sistem operasional sehari – hari, data disimpan berdasarkan aplikasi yang dibangun *individual* [OPR-12].

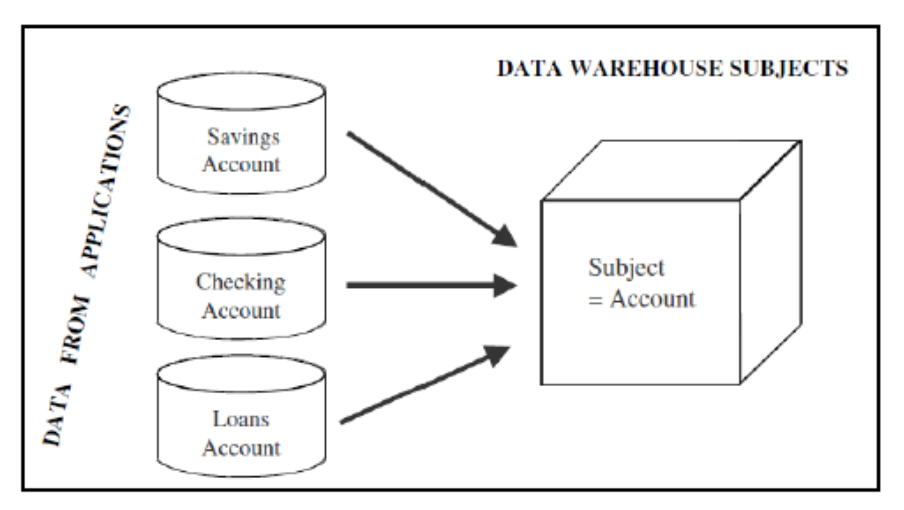


Gambar 2.4 Karakteristik Subject Oriented Data Warehouse
(Sumber : [PNP-01])

2. *Data Integrated*

Data warehouse memiliki karakteritik *Integrated* disini maksudnya adalah dimana *data warehouse* dapat melakukan penyimpanan dari berbagai data yang berasal dari sumber terpisah kedalam suatu format yang terintegrasi dan konsisten satu dengan lainnya [IWH-05]. Dari semua aspek *data warehouse*, integrasi data bisa menjadi salah satu aspek yang paling penting, karena *data warehouse* dapat memberikan gambaran fisik tunggal setelah proses integrasi tersebut [ALC-10]. Data yang dibutuhkan untuk *data warehouse* dapat berasal dari bermacam – macam sumber data yang terpisah dan harus dilakukan penghapusan terhadap data yang tidak konsisten dan melakukan

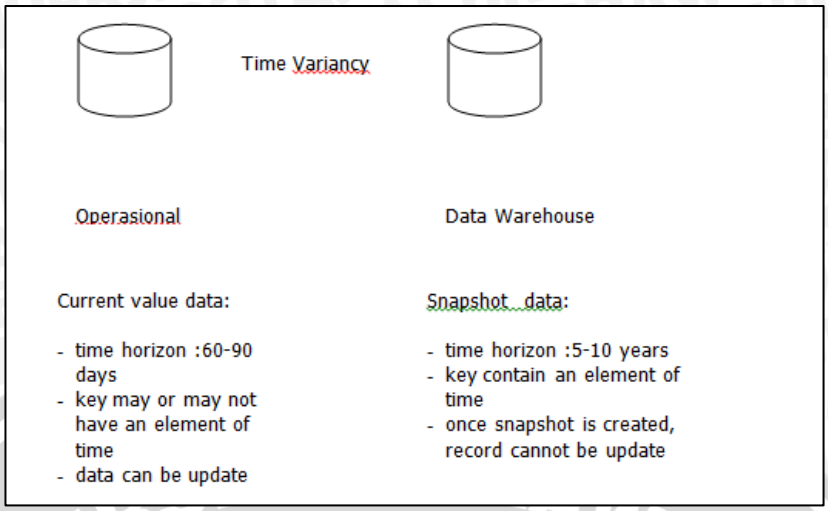
standardisasi pada berbagai elemen data. Ringkasnya, data yang digunakan pada *data warehouse* harus melalui transfomasi, konsolidasi, dan integrasi dari berbagai sumber data [PNP-01]. Data yang digunakan untuk *data warehouse* dapat berasal dari berbagai aplikasi yang berhubungan dengan keputusan yang diambil [OPR-12]. Sehingga dapat disimpulkan bahwa agar data pada *data warehouse* dapat saling terintegrasi, maka data yang berasal dari berbagai sumber harus melalui berbagai proses, seperti disusun ulang, konsolidasi, dirubah format (transformasi) dan diringkas serta seterusnya.



Gambar 2.5 Karakteristik Data Terintegrasi Data Warehouse
(Sumber : [PNP-01])

3. *Time Variant*

Pada karakteristik *time variant* di *data warehouse* adalah memungkinkan untuk dapat melakukan analisis masa lalu dan menghubungkannya dengan informasi saat ini serta prediksi masa depan [PNP-01]. Sehingga pada karakteristik ini dapat menjelaskan bahwa suatu dimensi waktu dapat menjadi hal yang vital agar dapat mengidentifikasi tren waktu sekarang dan memprediksi operasi – operasi kebutuhan masa depan [ALC-01]. Bisa disimpulkan, bahwa dimensi waktu bisa menjadi hal yang penting dalam pembentukan *data warehouse* agar mengetahui informasi data pada kurun waktu tertentu.

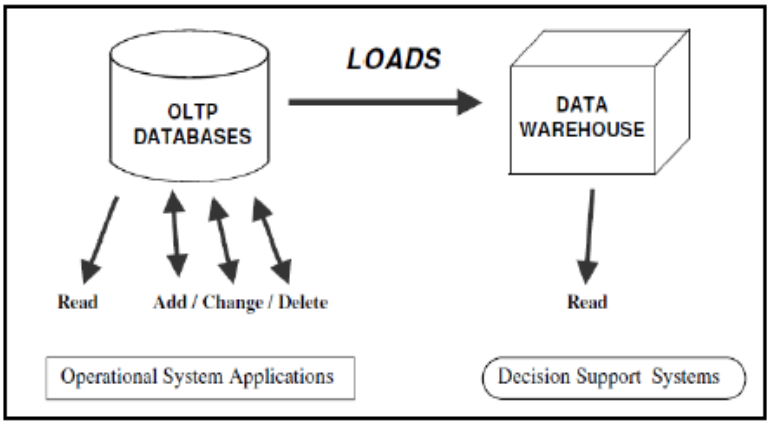


Gambar 2.6 Karakteristik Time Variance *Data Warehouse*

Sumber: [WCW-13]

4. *Non – Volatile*

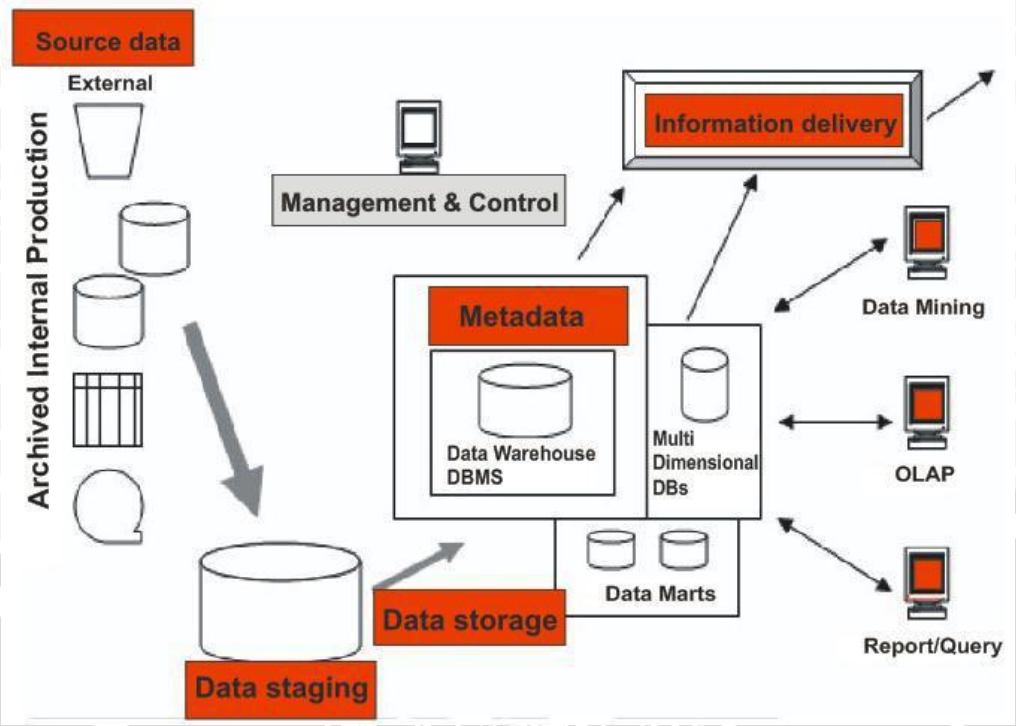
Data Warehouse memiliki karakteristik *non – volatile* disini maksudnya adalah bahwa *data warehouse* dirancang dan dibuat adalah bukan untuk menjalankan proses bisnis keseharian seperti pada OLTP [OPR-12]. Sehingga tidak ada pembaharuan data jika ada proses berjalan, namun bisa diperbarui dalam kurun waktu tertentu. Data yang ada pada *data warehouse* tidak selalu di-*update* setiap waktu seperti pada OLTP tapi akan dilakukan *refresh* dari sistem secara regular, data yang baru dapat ditambahkan sebagai tambahan bagi *data warehouse* tersebut daripada sebagai perubahan, sehingga data lama dapat tetap tersimpan. [IWH-05].



Gambar 2.7 Karakteristik *Non Volatile Data Warehouse*
(Sumber : [PNP-01])

2.8.2 ARSITEKTUR *DATA WAREHOUSE*

Arsitektur yang terdapat pada *data warehouse* dapat meliputi berbagai hal, dimulai dari data source yang dapat berupa operasional dari *database* maupun sumber dari luar (*eksternal*), kemudian juga terdapat alat untuk mengekstrak data, *transform* data, pembersihan data, integrasi data, *load* data ke *data warehouse*, dan secara periode kurun waktu tertentu dapat memperbarui *data warehouse* [OPR-12]. Dalam suatu arsitektur *data warehouse* juga dimungkinkan membuat suatu *data marts* yang disimpan dan diatur oleh satu atau bisa lebih *server* sehingga dapat menyajikan data secara multidimensional dalam bentuk atau format seperti *query*, grafik, tabel, penulisan laporan dan sebagainya, serta terdapat media penyimpanan dan mengatur *metadata* juga *tools* untuk memantau juga administrasi dari *data warehouse* [OPR-12].



Gambar 2.8 Arsitektur *Data Warehouse*

(Sumber : [PNP-01])

Arsitektur *data warehouse* memilliki berbagai pengembangan perancangan yang bervariasi, sehingga harus ditentukan model arsitektur yang cocok untuk dapat dibangun model rancangan *data warehouse* yang diinginkan. Seperti gambar 2.8, bahwa proses pengolahan data dari data sumber sebelum dimasukkan ke *data warehouse* melalui proses *data staging* terlebih dahulu. *Staging area* ini dibuat untuk dapat memudahkan proses integrasi data dari berbagai sumber dan pembersihan data agar didapat hasil data yang bagus dan berkualitas ketika dimasukkan ke *data warehouse*. Karena di *staging* area terdapat proses pembersihan data, penggabungan data dan standardisasi data [SDR-10]. *Data warehouse* memiliki beberapa komponen utama menurut *Connoly* dan *Begg* [CNB-2010]:

a. *Operational Data Source*

Merupakan sumber data yang didapat dari data operasional suatu perusahaan pada database transaksional (OLTP). *Operational data source*

ini dapat berasal dari berbagai sumber misalnya data operasional perusahaan, data yang disimpan dalam bentuk file, data pribadi, data pada sistem yang berada diluar perusahaan.

b. *Operational Data Store* (ODS)

Operational Data Store merupakan tempat penyimpanan data operasional perusahaan yang berguna untuk analisa.

c. ETL *Manager*

Tools yang dapat digunakan untuk melakukan proses ETL dari data *source* menuju *data warehouse.* ETL manager ini biasa disebut komponen *front – end*.

d. Warehouse Manager

Tools yang dapat digunakan untuk melakukan berbagai operasi yang berhubungan dengan manajemen data pada *data warehouse.*

e. Query Manager

Tools yang dapat digunakan untuk melakukan berbagai operasi yang terhubung dengan manajemen *query* dari seorang pengguna.

f. Detailed Data, Metadata, Lightly dan *Hightly Summarized Data*

Detailed data merupakan bagian dari *data warehouse* yang menyimpan berbagai rincian data pada skema *database. Metadata* merupakan bagian dari *data warehouse* yang digunakan untuk menyimpan definisi *metadata* yang berguna untuk proses dalam *data warehouse.* Biasanya metadata digunakan untuk beberapa tujuan misalnya, proses *ekstract* dan memasukkan data, manajemen *data warehouse*, dan proses manajemen *query*. *Lightly* dan *Hightly Summarized Data* merupakan bagian dari data *warehouse* yang digunakan untuk menyimpan data ringkasan yang berguna untuk mempercepat proses *query.*

g. Archive and Backup Data

Merupakan bagian dari *data warehouse* yang digunakan untuk menyimpan data detail dan rangkuman untuk diarsip atau di*backup*

h. End User Access tools

Tools ini dapat dibagi menjadi beberapa grup utama, antara lain adalah sebagai berikut:

a. *Reporting and Query Tools*

Merupakan *Tools* yang berguna untuk membuat dan menulis laporan. *Query tools* merupakan *tools* yang digunakan untuk melakukan *query* data pada *data warehouse*.

b. *Application Development Tools*

Merupakan tools yang dapat membantu *graphical data access tools* dan biasanya dikembangkan di lingkungan *client-server*.

c. OLAP *Tools*

Merupakan *tools* yang digunakan untuk mendukung pengguna agar dapat melakukan analisa data yang kompleks dengan banyak sudut pandang.

d. *Data Mining Tools*

Merupakan *tools* yang digunakan untuk melakukan *mining data*.

2.8.3 PERMODELAN *DATA WAREHOUSE*

Data Warehouse dan OLAP dibangun berdasarkan data model multidimensional. Pada permodelan ini diperlukan tabel fakta dan tabel dimensi.

2.8.3.1 TABEL FAKTA

Tabel fakta atau *Fact Table* merupakan representasi umum pada dimensional model di *database* relasional dan pada tabel fakta ini terdiri dari *foreign key* yang merupakan hasil relasi dengan tabel dimensi [KRM-02]. Tabel fakta adalah sebuah tabel yang berada di tengah dalam model multidimensi yang berhubungan dengan beberapa tabel dimensi [BNS-13]. Tabel fakta merupakan tabel yang mengandung angka dan data *history* serta terdiri dari *foreign key* yang merupakan hasil relasi dari beberapa tabel dimensi [IGR-06].

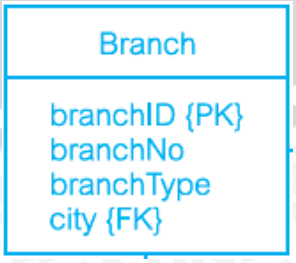
Dapat disimpulkan bahwa tabel fakta merupakan tabel utama yang ada pada *dimensional modeling* yang terdiri dari dua atau lebih *foreign key* dimana *key* tersebut terhubung dengan primary *key* pada tabel – tabel dimensi dan kolom yang menyimpan data agregasi berupa angka dimana kolom itu merupakan bentuk nilai pengukuran yang merupakan bentuk kesatuan data serta terdapat data *history*.

Gambar 2.9 *Fact Table*

(Sumber : [CNB-10])

2.8.3.2 TABEL DIMENSI

Tabel dimensi merupakan tabel dalam model multidimensi yang terdiri dari sebuah *primary key* dan atribut [KRM-02]. Dapat dikatakan, bahwa tabel dimensi merupakan tabel keterangan atau pelengkap bagi tabel fakta yang berisi penjelasan atau ringkasan data detail [IGR-06].



Gambar 2.10 *Dimension Table*

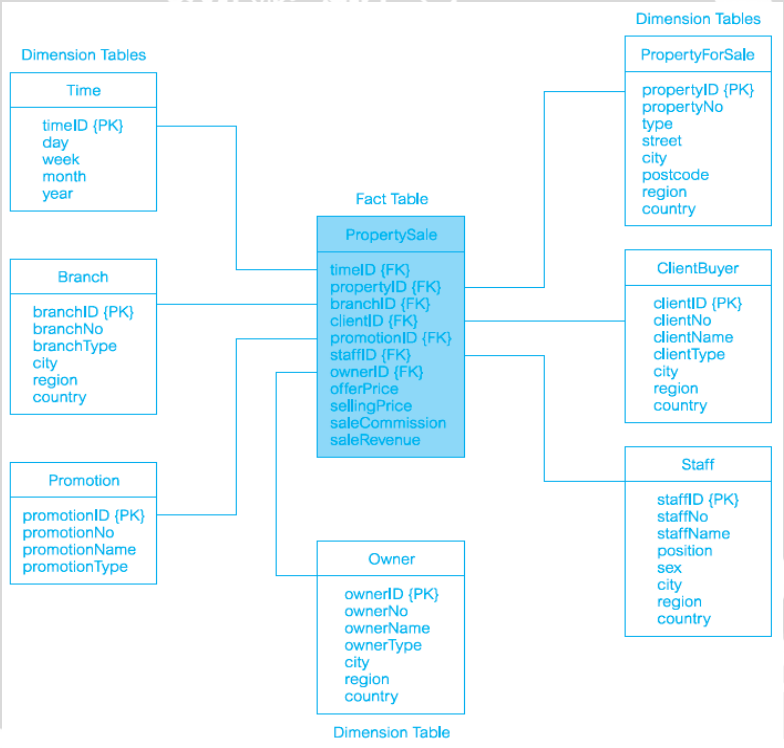
(Sumber : [CNB-10])

2.8.3.3 PENDEKATAN PERMODELAN *DATA WAREHOUSE*

Dalam permodelan *data warehouse*, ada beberapa pendekatan model yang digunakan dalam membangun *data warehouse*, yaitu :

a. *Star Schema*

Menurut Kimball, *star schema* dapat diartikan sebagai struktur logika yang mempunyai *fact table* yang ditempatkan di tengah dan dikelilingi oleh *dimension table* [KRR-10]. *Star Schema* merupakan teknik yang paling mudah dalam membangun suatu *data warehouse* [BNS-13]. Dapat disimpulkan *star schema* dapat menggambarkan hubungan antara tabel fakta dengan tabel dimensi dengan gambaran tabel fakta ditengah dan dikelilingi oleh beberapa tabel dimensi.



Gambar 2.11 *Star Schema*

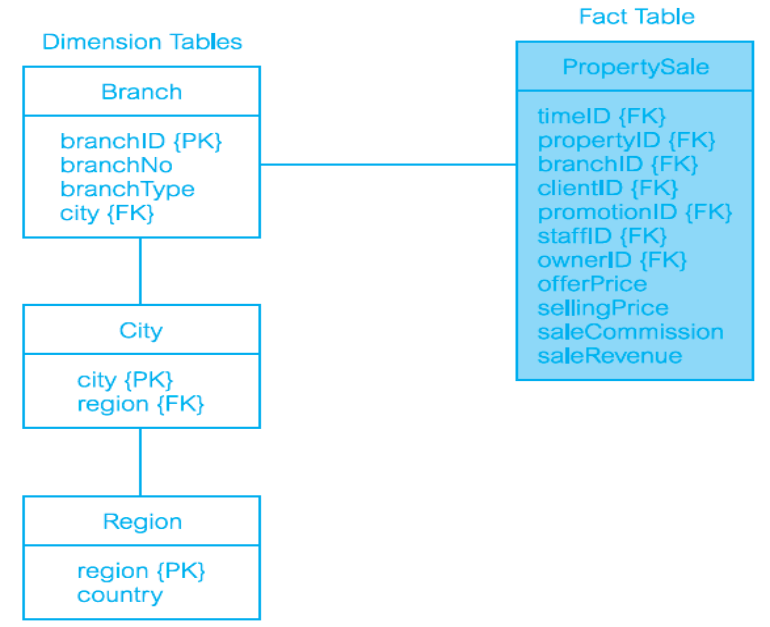
(Sumber : [CNB-10])

Star Schema memiliki beberapa kelebihan dan kekurangan, berikut adalah penjelasan dari kelebihan dan kekurangan dari *star schema* tersebut:

- Kelebihan Star Schema [ALC-10], [BNS-13] :
  - Waktu Respon yang cepat
  - Mudah Dipahami oleh pengguna
  - Mengurangi jumlah penggunaan join
  - Cocok untuk pemrosesan *query*
  - Mengoptimalkan proses navigasi
  - Pendesainan data secara parallel
- Kekurangan *Star Schema* [BAD-10]:
  - Ukuran penyimpanan yang besar
  - Tabel tidak ternormalisasi
  - Cukup sulit untuk melakukan *maintenance* data.

b. Snowflake Schema

Snowflake skema merupakan salah variasi dari *star schema* [IGR-06]. Menurut *Kimball*, skema *snowflake* merupakan salah satu pengembangan dari skema bintang yang memiliki perbedaan yaitu pada skema *snowflake* tabel dimensinya telah dinormalisasi dan adanya pengabungan antar tabel dimensi, sebelum tergabung dengan tabel fakta [KRM-10]. Jadi dapat disimpulkan bahwa skema *snowflake* ini dapat meminimalkan *redundancy* data yang mungkin terjadi pada *star schema*.



Gambar 2.12 *Snowflake Schema*

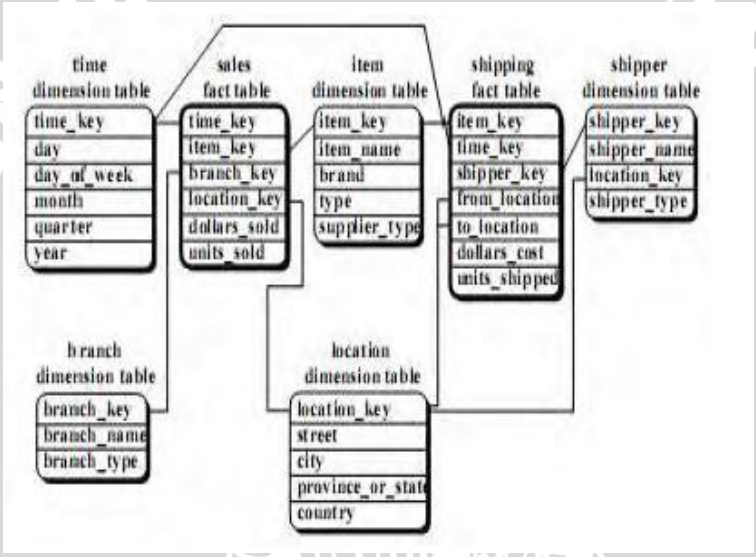
(Sumber : [CNB-10])

Snowflake Schema memiliki beberapa kelebihan dan kekurangan, berikut adalah penjelasan dari kelebihan dan kekurangan dari *snowflake schema* tersebut [BNS-13]:

- Kelebihan *snowflake Schema* :
  - Ukuran penyimpanan yang lebih kecil pada *database.*
  - Strukturnya yang sudah ternormalisasi sehingga memudahkan untuk *update* dan *maintenance*.
- Kekurangan *snowflake Schema*:
  - Dapat menyebabkan terjadinya penurunan performa pada *Query*, karena banyak join antar table
  - Terlalu kompleks sehingga terkadang menyebabkan sulitnya mencari isi schema dan sulit dimengerti pengguna.

c. *Galaxy Schema (fact constellation)*

Galaxy Schema merupakan skema gabungan dari *snowflake* skema dan *star schema*. *starflake schema (galaxy schema)* merupakan sebuah skema dengan struktur *hybrid* hasil gabungan antara *star schema* dan *snowflake schema* [CNB-10]. Dalam permodelan skema ini, beberapa tabel fakta berbagi tabel dimensi [SLD-13]. Sehingga dapat disimpulkan bahwa skema ini dapat dikatakan lebih kompleks karena melibatkan lebih dari satu tabel fakta, dan satu table dimensi dapat digunakan lebih dari satu tabel fakta.



Gambar 2.13 *Galaxy Schema*
(Sumber : [HJK-06])

Galaxy Schema memiliki beberapa kelebihan dan kekurangan, berikut adalah penjelasan dari kelebihan dan kekurangan dari *Galaxy schema* tersebut [IGR-06]:

- Kelebihan *Galaxy Schema*:
  - Efisien dalam hal pengaksesan data
  - Fleksibel terhadap perubahan dan pengembangan
  - Mudah adaptasi dengan kebutuhan user

- Kekurangan *Galaxy Schema*:
  - Sulit dalam pengelolaan karena desain yang kompleks dan rumit.

2.8.4 PERANCANGAN *DATA WAREHOUSE*

Metode perancangan yang digunakan untuk penelitian ini adalah metode yang didasarkan menurut *Kimball,* tentang metodologi Sembilan tahap dalam merancang *data warehouse* yang biasa disebut *Kimball Nine Step Methodology*. Sembilan langkah ini dilakukan untuk membangun *data warehouse* yang baik. Tahapan – tahapan yang dilakukan adalah sebagai berikut [CNB-05] :

1. **Pemilihan Proses**

Tahap awal ini merupakan tahap yang dilakukan untuk penentuan subjek permasalahan yang ingin diselesaikan, artinya proses yang dilakukan harus dapat menjawab pertanyaan bisnis yang telah disusun.

2. **Pemilihan *Grain***

Pada tahap ini yang harus dilakukan adalah memutuskan secara tepat *record* seperti apa yang ingin direpresentasikan dan digambarkan dalam tabel fakta.

3. **Identifikasi Dimensi**

Pada tahap ini yang dilakukan adalah menentukan dimensi yang digunakan untuk dapat mendetailkan penjelasan agar mudah dimengerti.

4. **Pemilihan Fakta**

Pada tahap ini yang dilakukan proses pemilihan fakta yang digunakan untuk hasil pada *data mart.* Dalam tabel fakta tersebut harus memiliki data agregasi yang nantinya akan ditampilkan dalam bentuk laporan bisa berupa grafik atau diagram.

5. **Penyimpanan *Pre-calculation* dalam tabel fakta**

Pada tahap ini yang dilakukan adalah memeriksa kembali pada tabel fakta apakah dapat dilakukan *pre-calculation*.

6. **Memastikan tabel dimensi**

Pada tahap ini yang dilakukan adalah memastikan tabel dimensi dengan cara menambahkan deskripsi atau gambaran teks pada tabel dimensi sehingga mudah dipahami pengguna.

7. **Memilih durasi *database***

Pada tahap ini yang harus dilakukan adalah menentukan batas waktu dari data sumber yang akan digunakan dan dipindahkan ke tabel fakta (*data warehouse*).

8. **Mengawasi Perubahan Dimensi**

Pada tahap ini yang harus dilakukan adalah mengawasi perubahan yang mungkin terjadi pada tabel dimensi (*Tracking Slowly changing dimensions*) Ada 3 cara yang dapat dilakukan untuk menanggulangi perubahan dimensi tersebut, yaitu yang pertama adalah mengganti langsung pada table dimensi, kedua adanya penambahan *record* baru apabila terjadi perubahan pada dimensi, dan yang ketiga adalah penambahan kolom baru yang berbeda apabila terjadi perubahan pada dimensi.

9. **Menentukan prioritas dan *Model query***

Pada tahap ini yang harus dilakukan adalah melihat pengaruh dari beberapa rancangan fisik, contohnya seperti mengurutkan tabel fakta, penyimpanan awal sebuah ringkasan atau agregasi, dan lain - lain.

2.9 ETL (EXTRACT, TRANSFORM, LOAD)

ETL merupakan singkatan dari tiga kata yang menggambarkan suatu proses, yaitu *Ekstract, Transform* dan *Load* [KNH-12]. Proses ETL dirancang dengan awalnya adalah ekstrak data dari sumber data, kemudian

integrasi data, lalu menggabungkan data dari banyak sumber yang terpisah agar dapat digunakan bersama yang akhirnya dapat memberikan data dengan format tertentu sehingga pengguna dapat membuat keputusan dari data hasil proses tersebut [TGJ-09]. ETL adalah proses mengambil semua data asli dari OLTP kemudian dilakukan proses pada data – data tersebut sehingga dapat dihasilkan sebuah hasil akhir dalam bentuk tabel, dimana pengguna dapat melakukan query pada tabel tersebut [BNS-13].

Menurut Kimball, ETL dibagi menjadi 3 proses besar [KRR-10]:

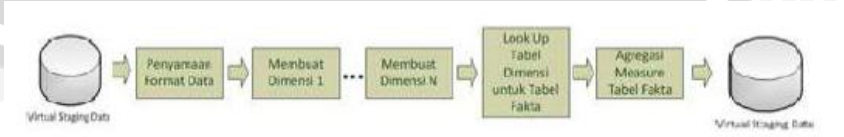
a. Extraction

Proses *Extraction* ini merupakan proses awal pada kegiatan ETL ini, dimana proses ini dilakukan untuk mengambil data – data yang diperlukan dari berbagai sumber data. Ringkasnya, proses yang dilakukan adalah ekstraksi data dari berbagai sumber data.

Gambar 2.14 *Proses Extract Data*
(Sumber:[KNH-12])

b. Transform

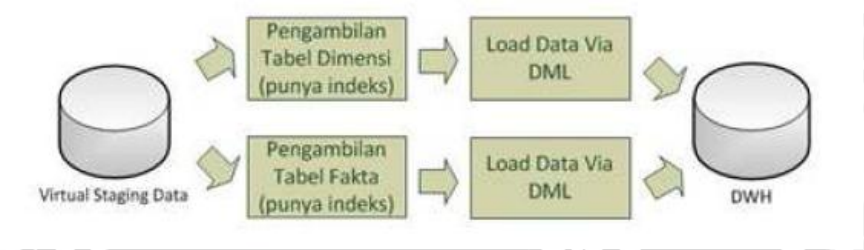
Proses yang dilakukan pada bagian ini adalah proses yang dilakukan setelah melalui proses *extraction*. Pada proses ini yang dilakukan adalah proses integrasi data seperti melakukan penyamaan format data dari berbagai sumber (melakukan standardisasi *domain* data).



Gambar 2.15 *Proses Transform Data*
(Sumber:[KNH-12])

c. Load

Setelah proses *extraction* dan *transform* dilakukan, maka proses terakhir dalam proses ETL ini yang dilakukan adalah *load*, yaitu proses untuk memasukkan data ke dalam data *warehouse* yang telah diimplementasikan.



Gambar 2.16 *Proses Load Data*

(Sumber:[KNH-12])

2.10 MATERIALIZED QUERY TABLE

Materialized Query Table adalah tabel yang didefinisikan berdasarkan dari hasil suatu *query*. Data yang terdapat didalam MQT didapat dari satu atau lebih tabel dimana yang menjadi tabel dasar dari MQT yang didefinisikan [IBD-13]. MQT dapat dipikirkan sebagai jenis dari *materialized view*, karena baik *view* maupun MQT didefinisikan dengan basis *query*. *Query* pada dasar *view* dijalankan sewaktu – waktu dimana *view* itu mengacu pada tabel dasar, sedangkan MQT benar – benar menyimpan hasil *query* sebagai data dan data tersebut dapat digunakan sebagai ganti dari data yang ada pada tabel dasar MQT [IBD-13]. MQT dapat dilakukan *maintained* secara otomatis oleh DB2 atau *maintained manual* dan *updated* oleh *user* maupun aplikasi [CMB-05].

Terdapat 2 *klausa control* kapan dan bagaimana MQT dibuat. *REFRESH DEFERRED* dapat diartikan bahwa data dapat berubah apabila *statement REFRESH TABLE* dilakukan. Sedangkan alternatif lain yaitu *REFRESH IMMEDIATE* akan menyebabkan perubahan secara langsung pada MQT bila terjadi perubahan pada tabel dasar [CMB-05].

```
CREATE TABLE bad_account
AS (SELECTstomer_name, customer_id, a.balance
cu FROM account a, customers c
WHERE status IN ('delinquent','problematic', 'hot')
AND a.customer_id = c.customer_id)
DATA INITIALLY DEFERRED
REFRESH DEFERRED );
```

Gambar 2.17 *Contoh Syntax MQT*

(Sumber:[CMB-05])

Ada beberapa pembatasan yang terdapat pada *materialized query table* [IBI-13]:

1. *Materialized Query Table* tidak dapat di *alter*.
2. *Base table* tidak dapat di*alter* pada bagian perubahan panjang kolom apabila tabel tersebut memiliki MQT.
3. Tidak dapat melakukan *import data* ke MQT.
4. Tidak dapat membuat *unique index* pada MQT
5. Tidak dapat membuat MQT dari *base table* pada hasil *query* yang bereferensi pada satu atau lebih *nickname*.

2.11 VERSIONING DATA

Versioning data merupakan salah satu fasilitas terbaru dari DB2, dimana ketika DB2 telah *upgrade* ke DB2 versi 10. Dengan dasar menyimpan sebuah sejarah perubahan data adalah sebuah tantangan, IBM memberi solusi sebagai berikut: Ketika baris diperbarui atau dihapus dari tabel dasar, maka DB2 10 akan menyisipkan baris diperbarui atau dihapus bersama dengan sistem *TIMESTAMP* ke tabel sejarah. Tabel sejarah implisit dibuat dengan tabel yang sama seperti tabel dasar tapi ditempatkan dalam partisi yang terpisah oleh *table space* [IBD-10]. Tabel sejarah berisi semua kolom dari tabel dasar, bersama dengan kolom yang mendukung audit dan pengelolaan baris [IBD-10].

DB2 mendukung dua jenis periode, yaitu periode sistem (*SYSTEM_TIME*) dan periode aplikasi (*BUSINESS_TIME*) [IBC-13].

Pada DB2 terdapat 3 Tipe dari *Temporal Tables* [MTN-12]:

1. ***System Period Temporal Tables*** **(STTs)**

   DB2 dengan jelas menyimpan *history* dari baris yang *diupdate* dan didelete setiap waktu.

2. ***Application Period Temporal Tables*** **(ATTs)**

   Aplikasi menyediakan tanggal atau *timestamps* untuk mendeskripsikan keabsahan bisnis dari data yang ada pada ATTs

3. ***Bitemporal Tables*** **(BTTs)**

   BTT mengatur kedua waktu baik *system time* dan *business time* dan menggabungkan semua kemampuan yang dimiliki oleh STTs dan ATTs.

Ketika mendefinisikan suatu tabel dasar untuk menggunakan *System Period Data Versioning*, atau mendefinisikan *System Period Data Versioning* pada tabel yang sudah ada, maka hal yang harus dilakukan adalah membuat tabel *History*, menentukan nama untuk tabel *History*, dan menciptakan *tablespace* untuk menjaga tabel tersebut. *Versioning* dapat didefinisikan dengan pernyataan *ALTER TABLE ADD VERSIONING* dengan *klausa USE HISTORY* TABLE [IBC-13].

2.12 IBM COGNOS INSIGHT DAN IBM DESIGN STUDIO

IBM *Cognos Insight* adalah solusi analisis milik personal yang memberi wewenang pada *user* untuk bebas mengeksplore, menganalisis, memvisualisasi, dan membagi data tanpa bergantung pada IT *assistance*. Manager Bisnis dan Analis dapat membuat dan membagi berbagai aplikasi, *dashboard*, dan visualisasi untuk membantu memecahkan tantangan individu maupun kelompok [IBS-13].

IBM *Design Studio* adalah komponen yang dapat digunakan untuk membuat *project data warehouse* dari *Relational Database* DB2. *Design Studio* dapat digunakan untuk membantu melakukan *data movement*, *data mining* dan OLAP [IBV-13].

2.13 DB2BATCH TOOL

Db2batch adalah *tools benchmark* yang dapat digunakan pada *command line*. *Tools* ini dapat membaca, menyiapkan, dan mendeskripsikan *SQL Query* [IBV-13]. *Tools* ini juga dapat digunakan untuk pengecekkan performa *query* dengan *line* seperti dibawah ini.

```
db2batch -d dbname -f input.txt -r output.txt
```

Gambar 2.18 *Contoh Syntax DB2batch*

(Sumber : [IBV-13])

Dbname dimasukkan dengan nama *database* dari asal *query*. Input dan output teks bisa dimasukkan berdasarkan nama yang diinginkan [IBV-13].