BAB 2
TINJAUAN PUSTAKA

2.1 Profil Perusahaan

CV Karya Anugerah Tritunggal didirikan pada tanggal 31 Juli 2007 dengan nomor SIUP 510/1-6896/2009/P.7/2246-BPPT dan beralamat di Jalan Cikawao Permai B8, Bandung. CV Karya Anugerah Tritunggal dibentuk atas landasan kepercayaan kepada Tuhan Yang Maha Kuasa untuk turut berperan serta dalam meningkatkan kualitas hidup bangsa melalui peningkatan kualitas sumber daya manusia dan pemanfaatan sumber daya alam yang optimal.

Desentralisasi kekuasaan pemerintah dalam bidang pertambangan memberikan peluang pada perusahaan untuk melengkapi pelaku tambang di Indonesia, terutama dalam bidang perdagangan batu bara skala menengah-kecil. Kegiatan usaha perusahaan adalah mengambil keuntungan dari pengepakan batu bara yang telah ditambang oleh perusahaan lain, dengan basis pembelian di Pelabuhan Cirebon. CV Karya Anugerah Tritunggal menyediakan tiga jenis dari kualitas dan spesifikasi batubara, dua jenis batu bara dengan total sulfur kurang dari 1% dan satu jenis dengan nilai kalori berkisar 4000 kg.

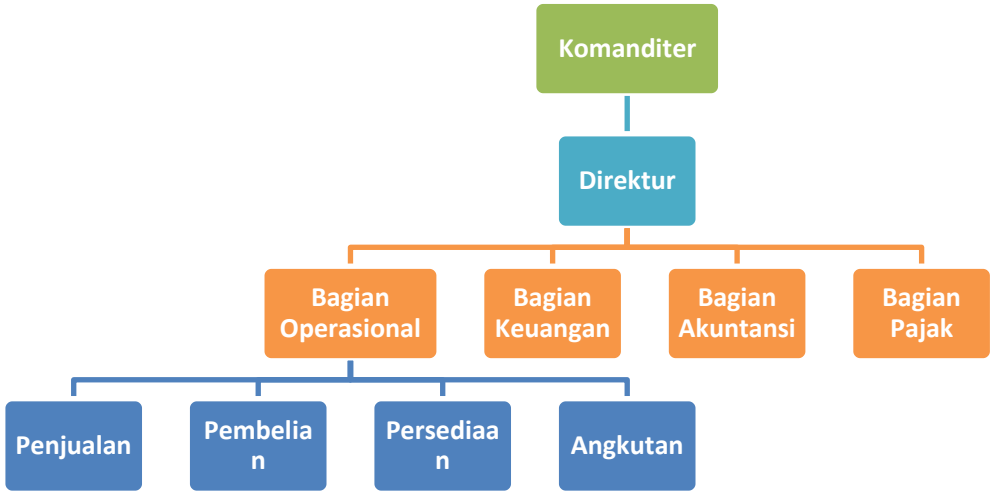
Visi CV Karya Anugerah Tritunggal adalah menjadi pedagang batu bara yang efisien dan handal. Visi tersebut dapat dicapai dengan melaksanakan misi sebagai berikut:

1. Memberikan pelayanan terbaik bagi *costumer* dalam bidang perdagangan batu bara.
2. Mengembangkan sumber daya manusia untuk mencapai kesejahteraan yang lebih baik.
3. Meningkatkan kompetensi perusahaan sehingga mencapai efisiensi yang optimum dengan teknolgi yang mutakhir.

2.2 Struktur Organisasi Perusahaan

CV. Anugerah Karya Tritunggal memiliki susunan organisasi dan pembagian kerja untuk melaksanakan kegiatan bisnis dan tugas pokok.

Perusahaan ini dikepalai oleh seorang komanditer dan dibantu oleh direktur. Bagan struktur organisasi CV. Anugerah Karya Tritunggal akan ditampilkan pada gambar 2.1.



Gambar 2.1 Bagan Susunan Organisasi CV. Karya Anugerah Tritunggal, 2015
(Sumber: Olah Data CV. Anugerah Karya Tritunggal, 2015)

2.2.1 Deskripsi Kerja

Berikut ini adalah penjelasan ringkas mengenai tugas dan tanggung jawab tiap bagian kerja di CV. Anugerah Karya Tritunggal.

1. Komanditer

   Komanditer bertugas untuk mengawasi jalannya kegiatan perusahaan dan menerima laporan perkembangan perusahaan dari direktur.

2. Direktur

   Bertugas sebagai koordinator dan mengendalikan semua kegiatan dalam perusahaan, seperti merencanakan dan mengembangakan sumber pendapatan, pembelanjaan dan kekayaan perusahaan.

3. Bagian Operasional

   Bagian ini bertugas dalam pengambilan keputusan yang berkaitan dengan fungsi operasi dan sistem transformasi, perancangan, dan desain sistem,

serta rancangan tugas pekerjaan. Bagian operasional membawahi empat sub-bagian yaitu penjualan, pembelian, persediaan, dan angkutan.

a. Penjualan

   Sub-bagian penjualan bertugas untuk mencatat pesanan pembelian (*purchase order*) dari konsumen, membuat faktur penjualan (*invoice*) dan surat jalan, melakukan penagihan kepada konsumen, dan membuat laporan penjualan.

b. Pembelian

   Sub-bagian agian ini bertugas untuk memesan batu bara kepada pihak *supplier*, melakukan konfirmasi pembelian, mencatat transaksi pembelian, dan membuat laporan pembelian, serta melakukan pembayaran hutang pembelian.

c. Persediaan

   Sub-bagian persediaan memiliki tugas untuk mencatat arus masuk dan arus keluar setiap batu bara.

d. Angkutan

   Tugas dari sub-bagian ini adalah melakukan konfirmasi kepada bagian persediaan, mengirim batu bara kepada konsumen, membuat faktur angkutan, dan melakukan pembayaran tagihan angkutan kepada pihak ketiga.

4. Bagian Keuangan

   Bagian keuangan bertanggung jawab atas seluruh kegiatan keuangan perusahaan termasuk melakukan perencanaan keuangan, mengumpulkan dana perusahaan dan mengamankan dana tersebut.

5. Bagian Akuntansi

   Tugas bagian akuntansi adalah untuk menangani setiap masalah keuangan dan membuat laporan keuangan.

6. Bagian Pajak

   Bagian ini bertugas untuk mengatur dan menghitung pajak perusahaan.

2.3 Landasan Teori

Landasan teori membahas mengenai materi atau teori apa saja yang digunakan sebagai acuan dalam membuat tugas akhir ini. Landasan teori yang diuraikan merupakan hasil studi literatur, baik dari buku, maupun situs internet.

2.3.1 Data

Menurut O'Brien, data adalah fakta-fakta atau obeservasi mengenai fenomena fisik atau transaksi bisnis [1]. Lebih khusus lagi, data adalah ukuran objektif dari atribut (karakteristik) dari entitas seperti orang-orang, tempat, benda atau kejadian.

2.3.2 Informasi

Informasi adalah hasil analisis dan sintesis terhadap data. Dengan kata lain, informasi dapat dikatakan sebagai data yang telah diorganisasikan ke dalam bentuk yang sesuai dengan kebutuhan seseorang. Menurut Encyclopedia of Computer Science and Engineering, banyak ilmuwan di bidang informasi menerima definisi standar bahwa informasi adalah data yang digunakan dalam pengambilan keputusan. Alasanya adalah bahwa informasi bersifat relatif, relatif terhadap situasi, relatif terhadap waktu saat keputusan diambil, juga relatif terhadap pembuat keputusan, dan bahkan juga relatif terhadap latar belakang pengambil keputusan.

Segala sesuatu yang dianggap penting pada suatu waktu bisa saja tidak berguna pada waktu yang lain. Ada kemungkinan pula bahwa sesuatu yang dianggap penting oleh seorang pengambil keputusan tidak dianggap penting bagi orang lain.

Dalam era reformasi, informasi menjadi sumber penting untuk melakukan pengambilan keputusan. Informasi dapat mengurangi ketidakpastian dan mempermudah pengambilan keputusan.

2.3.3 Pengertian Basis Data (*Database*)

Menurut O'Brien, *database* adalah kumpulan terpadu dari elemen data logis yang saling berhubungan [1]. *Database* mengonsolidasi banyak catatan yang sebelumnya disimpan dalam file terpisah agar kelompok data yang sama menyediakan banyak aplikasi.

2.3.4 *Database Management System* (DBMS)

Menurut Connolly dan Begg, "*Database Management System* (DBMS) is a software system that enables users to define, create, maintain, and control access to the *database*" [2]. Yang dapat diartikan bahwa DBMS adalah sebuah system software yang memungkinkan user untuk mendefinisikan, membuat, memelihara, dan mengawasi akses ke *database*. Sebuah DBMS harus menyediakan fasilitas sebagai berikut :

1. Mampu mendefinisikan *database*, biasanya melalui *Data Definition Language* (DDL). DDL memungkinkan user untuk menentukan tipe data, struktur, dan batasan terhadap data yang akan disimpan ke *database*.
2. Memungkinkan user untuk memasukan (*insert*), merubah (*update*), menghapus (*delete*), dan mengambil (*retrieve*) data dari *database*, biasanya melalui *Data Manipulation Language* (DML). DML memungkinkan user untuk melakukan query.
3. Menyediakan kendali akses ke *database*. Sebagai contoh DBMS dapat menyediakan :
    a. Sistem keamanan yang memungkinkan untuk mencegah user yang tidak berkepentingan untuk mengakses *database*.
    b. Sistem integrasi yang menjaga konsistensi data yang tersimpan.
    c. Sistem kendali yang memungkinkan *database* untuk diakses secara bersamaan.
    d. Sistem pemulihan yang memungkinkan untuk mengembalikan keadaan *database* ke kondisi konsisten yang sebelumnya jika terjadi kesalahan, termasuk kesalahan hardware dan software.

e. Sebuah katalog yang bisa diakses oleh user yang di dalamnya terdapat deskripsi atau penjelasan dari data yang terdapat di dalam *database*.

2.3.5 *Data warehouse*

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

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* [3].

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 [3] .

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 [3].

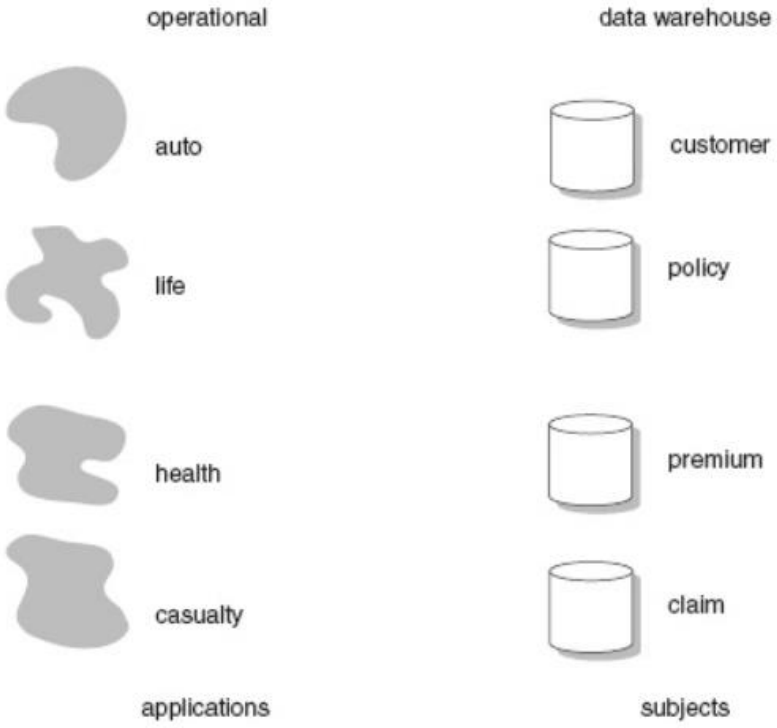
Dari definisi-definisi yang dijelaskan tadi, dapat disimpulkan *data warehouse* adalah *database* yang saling bereaksi yang dapat digunakan untuk *query* dan analisisis, bersifat orientasi subjek, terintegrasi, *time-variant*, tidak berubah yang digunakan untuk membantu para pengambil keputusan.

2.3.5.1 Karakteristik *Data warehouse*

Menurut Inmon, *Data warehouse* didefinisikan dengan karakteristik sebagai berikut [3] :

1. *Subject Oriented* (Berorientasi Subjek)

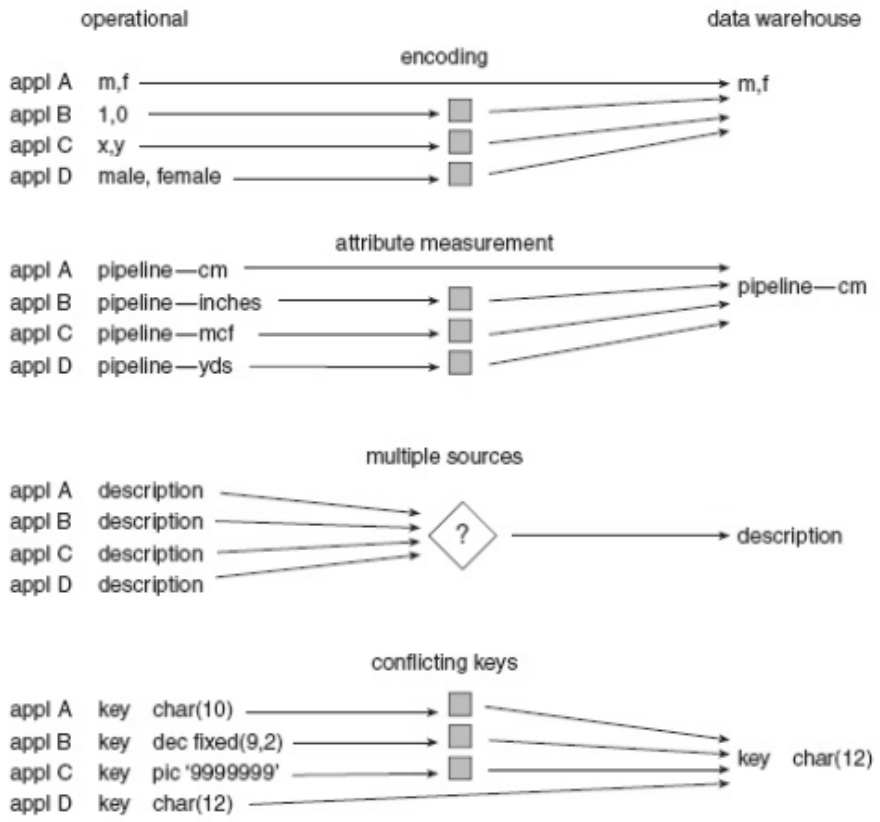
   Subject oriented berarti *data warehouse* dibuat atau disusun berdasarkan pada subjek utama dalam lingkungan perusahaan, bukan berorientasi pada proses atau fungsi aplikasi seperti yang terjadi pada lingkungan operasional. Sebagai contoh adalah sebuah perusahaan asuransi aplikasi terdiri dari mobil, kesehatan, jiwa, dan kehilangan. Sedangkan pada *data warehouse* diatur berdasarkan pelanggan, polis, premi dan klaim.



Gambar 2.2 *subject oriented* dalam *data warehouse* [3]

2. *Integrated* (Terintegrasi)

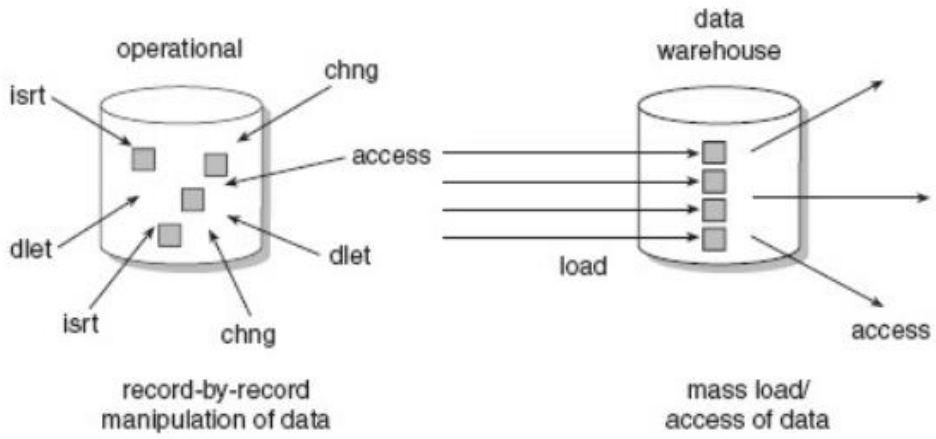
   Data dalam *data warehouse* bersifat terintegrasi karena bersumber dari sistem – sistem aplikasi yang berbeda dalam perusahaan. Sumber data demikian sering tidak konsisten, misalnya karena berbeda format. Sumber data yang terintegrasi ini harus dibuat konsisten untuk memberikan data yang seragam pada para pengguna.



Gambar 2.3 *Integrated* data dalam *data warehouse* [3]

3. *Non Volatile*

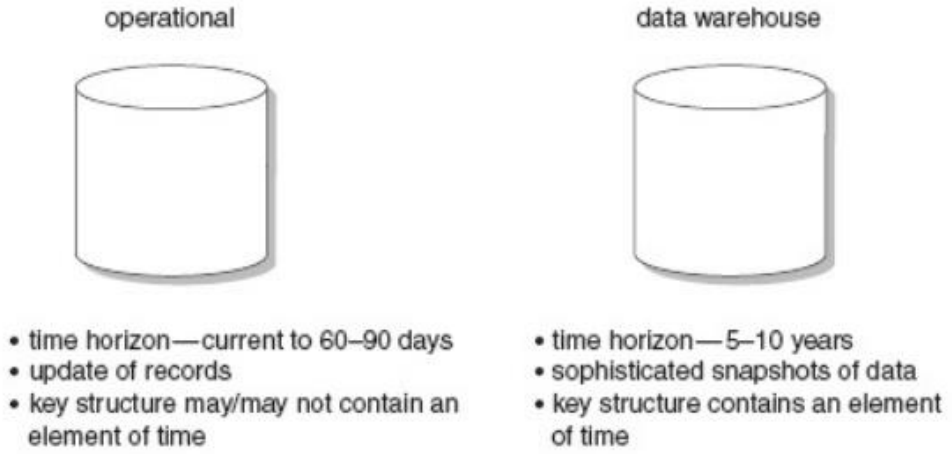
   Data dalam *data warehouse* tidak di-update dalam real time melainkan diperbarui secara periodik dari sistem operasional. Data baru selalu ditambahkan sebagai tambahan bagi *database*, bukan sebagai pengganti. *Database* secara terus–menerus mengambil data baru, menambahnya, dan mengintegrasikannya dengan data sebelumnya.



Gambar 2.4 *non-volatility* dalam *data warehouse* [3]

4. *Time Variant* (Rentang Waktu)

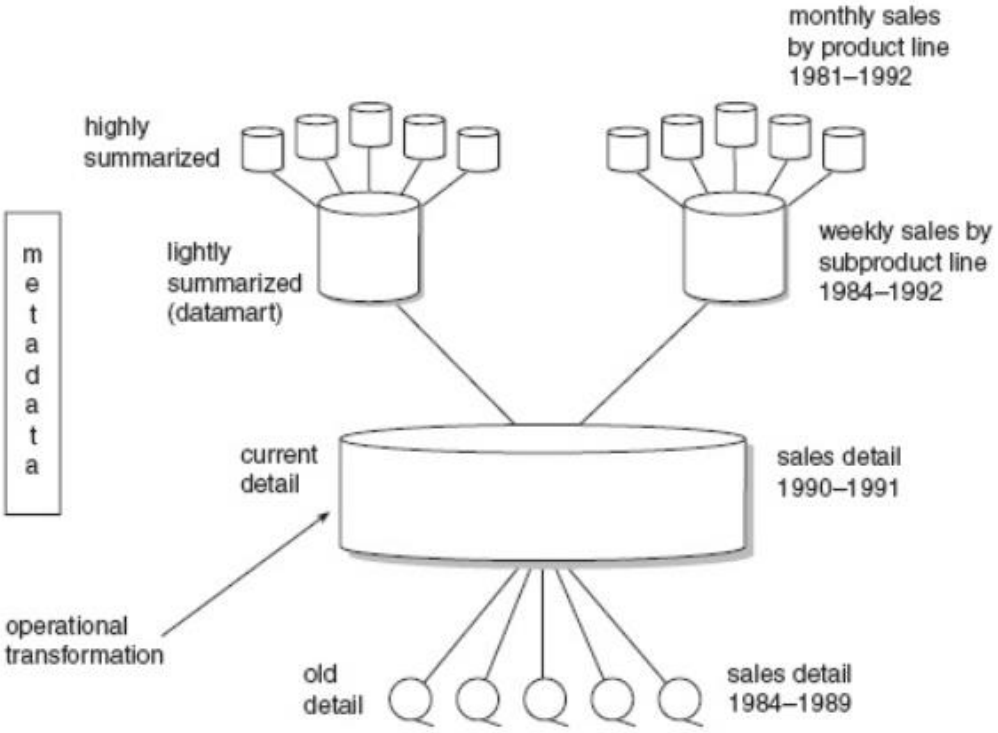
   Data dalam *data warehouse* bersifat tepat dan valid dalam jangka waktu tertentu. Data dalam *data warehouse* terdiri dari serangkaian snapshot, masing–masing menunjukan data operasional yang diambil pada suatu waktu tertentu.



Gambar 2.5 *time variancy* dalam *data warehouse* [3]

2.3.5.2 Struktur *Data warehouse*

Struktur *data warehouse* menunjukan level detail yang berbeda dalam *data warehouse*. Terdapat *older level of detail, current level of detail, level of lightly summarized* data (level *data mart*), dan *level of highly summarized data*. Data mengalir kedalam *data warehouse* dari lingkaran operasional. Biasanya transformasi penting dari data terjadi pada jalur dari level data operasional ke level *data warehouse* [3].



Gambar 2.6 Struktur dari data warehouse [3]

Sekali data disimpan, data melalui *current detail* ke *older detail*. Selama data diringkas, data melalui *current detail* ke *older detail*. Selama data diringkas, data melalui *current detail* ke *lightly summarixed data*, kemudian dari *lightly summarized data* ke *highly summarized data*.

1. *Current Detail Data*

   Current detail data adalah datadetail yang sedang aktif saat ini, mencerminkan keadaan yang sedang berjalan saat ini dan merupakan tingkat terendah dalam *data warehouse*. *Current detail* data ini biasanya memerlukan media penyimpanan data yang cukup besar. Alasan perlu diperhatikan *current detail* data adalah sebagai berikut :

   a. Menggambarkan kejadian yang baru terjadi dan selalu menjadi perhatian utama.
   b. Hampir selalu disimpan di media penyimpanan karena cepat diakses tetapi mahal dan kompleks dalam pengaturannya.
   c. Dapat digunakan dalam membuat rekapitulasi sehingga current detail data harus akurat.
   d. Jumlahnya sangat banyak dan disimpan pada tingkat penyimpanan rendah.

2. *Old Detail Data*

   Old detail data adalah data historis dapat berupa hasil *back-up* yang dapat disimpan dalam media penyimpanan yang terpisah dan dapat diakses kembali pada saat tertentu. Data ini jarang diakses sehingga disimpan dalam media penyimpanan alternatif seperti tape dan disk. Data ini biasanya memiliki tingkat frekuensi akses yang rendah. Penyusunan file atau directory dari data ini disusun berdasarkan umur dari data yang bertujuan mempermudah untuk pencarian atau pengaksesan kembali.

3. *Lightly S ummarized Data*

   Data ini merup akan ringkasan atau rangkuman dari *current detail data*. Data ini dirangkum berdasarkan periode atau dimensi lainnya sesuai dengan kebutuhan. Ringkasan dari current detail data belum bersifat total summary. Data-data ini memiliki detail tingkatan yang lebih tinggi dari *current detail data* dan mendukung kebutuhan warehouse pada tingkat departemen. Tingkatan data ini disebut juga dengan data mart. Akses terhadap data jenis ini banyak digunakan untuk View suatu kondisi yang sedang atau sudah berjalan.

4. *Highly Summarized Data*

   Data ini merupakan tingkat lanjutan dari *lightly summarized data*, yang merupakan hasil ringkasan yang bersifat totalitas, dapat diakses misalnya untuk melakukan analisis perbandingan data berdasarkan urutan waktu tertentu dan analisis menggunakan data multidimensi.

5. *Metadata*

   Metadata adalah data mengenai data yang berisi lokasi dan deskripsi sistem komponen *data warehouse* seperti nama, definisi, struktur, dan isi dari *data warehouse*, identifikasi dari sumber data yang berwenang, dan lainnya *Metadata* merupakan data yang menjelaskan tentang data dan merupakan suatu bentuk jaringan yang sangat penting bagi penggunaan *data warehouse*. *Metadata* dibuat untuk menjawab kebutuhan dari suatu fungsi tertentu karena setiap departemen biasanya menggunakan struktur data yang spesifik meskipun sumber datanya sama. Peranan *Metadata* yaitu :

   a. Sebagai directory untuk membantu penggunaan *data warehouse* menempatkan isi data dan mengetahui lokasi data dalam *data warehouse*.
   b. Sebagai panduan untuk menempatkan (*mapping*) data pada saat data ditransformasikan dari OLTP ke dalam lingkungan *data warehouse*.
   c. Sebagai panduan untuk menghasilkan rangkuman dari *current detailed data* menjadi *lightly summarized data* dan dari *lightly summarized data* menjadi *highly summarized data*.

2.3.5.3 Arsitektur *Data warehouse*

Menurut Connolly, komponen–komponen utama sebuah *data warehouse* antara lain [2]:

1. *Operational Data*

   Data untuk *data warehouse* berasal dari :

   a. Mainframe data operasional yang berada pada tingkatan *database* generasi pertama dan *database* jaringan. Diperhatikan sebagian besar data operasional perusahaan disimpan pada sistem tersebut.
   b. Data departemen yang berada di sistem file DBMS relasional (seperti SQL Server 2008).
   c. Data pribadi yang berada di server dan workstation pribadi.
   d. Sistem–sistem eksternal seperti internet, *database* yang tersedia secara komersil, dan *database* yang berhubungan dengan pemasok atau pelanggan perusahaan.

2. *Operational Datastore*

   Sebuah *operational data store* (ODS) adalah sebuah tempat penyimpanan data operasional terkini dan terintegrasi, yang digunakan untuk kebutuhan analisis. ODS memiliki struktur dan sumber data yang sama dengan *data warehouse*, berperan sebagai tempat penyimpanan data sebelum dipindahkan ke *data warehouse* ODS menyimpan data yang telah di-extract dari sistem sumber dan telah dibersihkan. Dengan demikian, proses pengintegrasian dan rekontruksi data untuk *data warehouse* menjadi lebih sederhana.

3. *Load Manager*

   Load Manager melakukan semua operasi yang berhubungan dengan fungsi extracting / mengambil data dan fungsi loading / meletakkan data ke dalam *data warehouse*. Data dapat di-extract dari sumber – sumber data atau pada umumnya diambil dari *operational data store*. Operasi yang dilakukan *Load Manager* dapat berupa transformasi data yang sederhana untuk mempersiapkan pemasukan data ke dalam *data warehouse*. Ukuran dan kompleksitas komponen ini akan berbeda sesuai dengan *data warehouse* yang dirancang dan dapat dibangun menggunakan kombinasi antara vendor loading dan custom- build programs.

4. *Warehouse Manager*

   Warehouse manager melakukan semua operasi yang berhubungan dengan manajemen data dalam *data warehouse*. Komponen ini dibangun menggunakan vendor dan management tools dan custom-build programs. Operasi yang dilakukan oleh *data warehouse* manager berupa :

   a. Melakukan analisis data untuk menjaga konsistensi data.
   b. Melakukan transformasi dan penggabungan sumber data dari penyimpanan sementara ke dalam table – table *data warehouse*.
   c. Menciptakan index dan view pada base tables.
   d. Melakukan denormalisasi (jika diperlukan).
   e. Melakukan agregasi (jika diperlukan)
   f. Melakukan back-up dan archive / back-up data

5. *Query Manager*

   Query manager melakukan semua operasi yang berhubungan dengan management user queries. Komponen ini dibangun menggunakan vendorend-user data *access tools*, *data warehouse* monitoring tools, fasilitas *database*, dan custom build-in programs. Kompleksitas queries manager ditentukan oleh fasilitas yang disediakan end-user access tools dan *database*. Operasi yang dilakukan komponen ini berupa pengarahan query pada table – table yang tepat dan penjadwalan eksekusi query.

6. *Detailed Data*

   Komponen ini menyimpan semua detil dalam skema basis data. Pada umumnya beberpa data tidak disimpan secara fisik, tetapi dapat dilakukan dengan cara agregasi. Secara periodik data detil ditambahkan ke *data warehouse* untuk mendukung agregasi data.

7. *Lighly and Highly Summarized Data*

   Komponen ini menyimpan semua data yang diringkas oleh warehouse manager. Data perlu diringkas dengan tujuan untuk mempercepat performa query. Ringkasan data terus diperbaharui seiring dengan adanya data yang baru yang masuk ke dalam *data warehouse*

8. *Archive / Backup Data*

   Komponen ini menyimpan data detil dan ringkasan data dengan tujuan untuk menyimpan dan backup data. Walaupun ringkasan yang diperoleh dari data mendetil, ringkasan perlu di backup juga apabila data tersebut disimpan melampaui periode penyimpanan data detil. Data kemudian dipindahkan ke media penyipanan seperti magnetic tape atau optical disc.

9. *Meta-data*

   Komponen ini menyimpan semua definisi meta-data (informasi mengenai data) yang digunakan dalam proses dalam *data warehouse*. Meta-data digunakan untuk berbagai tujuan, antara lain :

   a. Proses extracting dan loading, meta-data digunakan untuk memetakan sumber data dalam *data warehouse*.
   b. Proses manajemen warehouse, meta-data digunakan untuk menghasilkan tabel ringkasan (summarized tables).
   c. Sebagai bagian dari proses manajemen query meta-data digunakan untuk mengarahkan sebuah query pada sumber data yang tepat.

10. *End - User Access Tools*

    Tujuan utama dari *data warehouse* adalah dengan menyediakan informasi bagi pengguna untuk pembuatan keputusan yang strategis dalam berbisnis. Para pengguna berinteraksi dengan *data warehouse* menggunakan end-user access tools. Berdasarkan kegunaan *data warehouse*, terdapat lima kategori *end-user access tools*, yaitu :

    a. *Reporting and Query Tools*

       Reporting tools meliputi production reporting tools dan reports writers. Production reporting tools digunakan untuk menghasilkan laporan operasional secara berkala. Query tools untuk relasional *data warehouse* dirancang untuk menerima SQL atau menghasilkan SQL statements untuk proses query data yang tersimpan di warehouse.

b. *Application Development Tools*

Application development tools menggunakan *graphical data acsess tools* yang dirancang khusus untuk lingkungan client-server. Beberapa aplikasi perlu diintegrasikan dengan OLAP tools, dan dapat mengakses semua sistem basis data utama.

c. *Executive Information System (EIS) Tools*

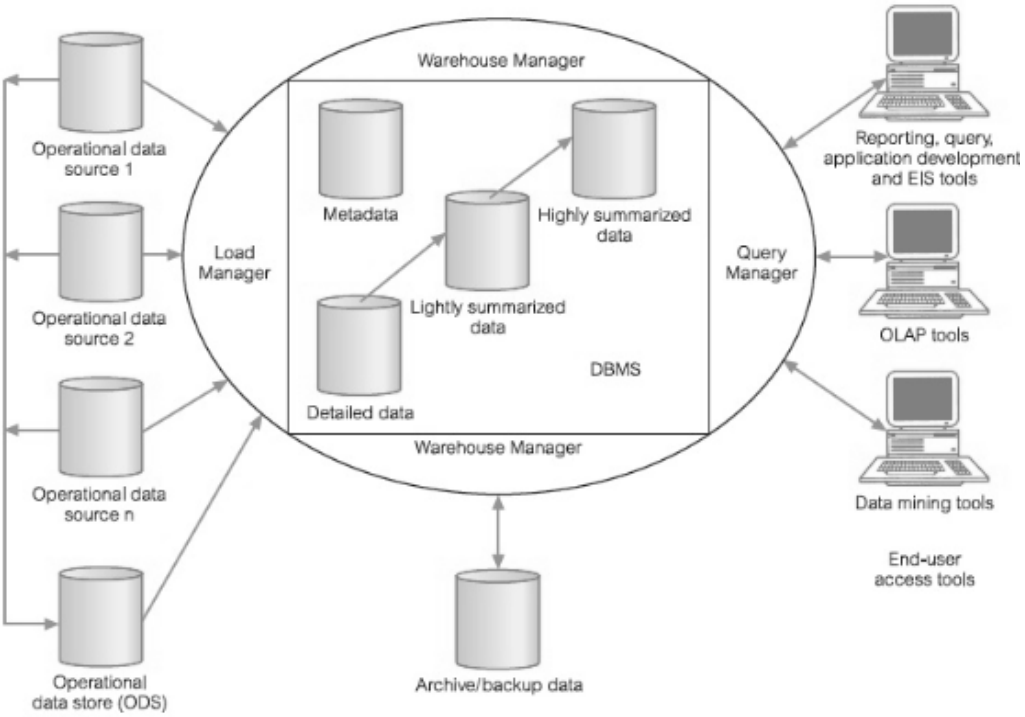
EIS sering dikenal sebagai '*everyone's information systems*' (sistem informasi setiap orang). Awalnya dikembangkan untuk mendukung pembuatan kebutuhan top level yang strategis. Akan tetapi, kemudian meluas mendukung semua tingkat manajemen. EIS tools pada awalnya berhubungan dengan mainframes yang memungkinkan para pengguna untuk membangun aplikasi pendukung keputusan yang bersifat grafik untuk menyediakan sebuah overview mengenai data perusahaan dan akses pada sumber data eksternal. Kini, EIS banyak dilengkapi dengan fasilitas query dan menyediakan custom build applications untuk area bisnis seperti penjualan, pemasaran, dan keuangan.

d. *Online Analytical Processing (OLAP) Tools*

OLAP tools didasarkan pada konsep basis data yang bersifat multi-dimensi dan memperbolehkan pengguna untuk menganalisis data dari sudut pandang yang kompleks dan multidimensi. Alat bantu ini mengasumsikan bahwa data diatur dengan model multi-dimensi yang khusus (MDDB) atau sebuah relational basis data yang dirancang untuk memungkinkan query multi-dimensi.

e. *Data Mining Tools*

Data mining adalah proses menemukan kolerasi, pola, dan tren yang baru yaitu dengan melakukan penggalian pada sejumlah data menggunakan teknik statistik, matematis, dan *artificial intelligenten* (AI). Data mining memiliki potensi untuk menggantikan kemampuan OLAP tools



Gambar 2.7 Arsitektur *data warehouse* [2]

2.3.5.4 Proses ETL (*Extraction, Transformation, Loading*)

Extraction, Transformation, and Loading (ETL) memiliki peranan utama dalam *data warehouse*. ETL juga merupakan suatu komponen utama untuk penyukses *data warehouse* yang dikembangkan. ETL merupakan suatu terminology umum yang digunakan dalam *data warehouse* yang memiliki proses mengekstrak data dari sumber system, mengubahnya berdasarkan kebutuhan bisnis dan menyajikannya ke dalam sebuah *data warehouse*. ETL menarik data dari berbagai sumber data dan menaruhnya ke dalam sebuah *data warehouse*. ETL proses bukanlah proses yang dilakukan sekali, tetapi secara periodik/memiliki jadwal seperti bulanan, mingguan, harian, bahkan dalam hitungan jam. ETL merupakan suatu kombinasi kompleks dari proses dan teknology yang mengkonsumsi sebagian besar usaha pengembangan *data warehouse* dan membutuhkan kemampuan dari *Business Analysts*, *Database*

Deasigners dan *Application Developer* [4]. Framework ETL memiliki 3 proses utama yaitu Extraction, Transformation, dan Loading [4]

a. *Extraction*

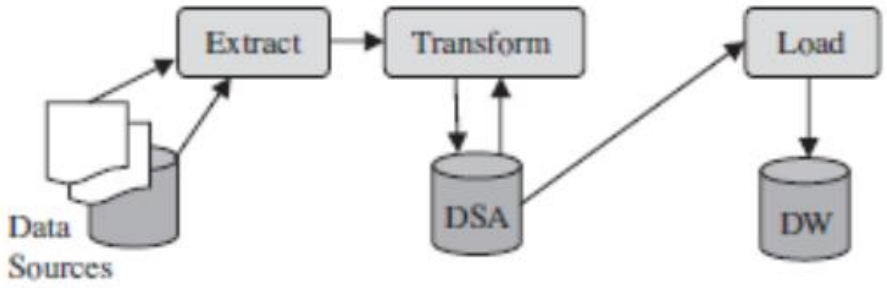
   Langkah pertama dalam skenario ETL dengan mengekstrak data yang terdapat dalam sumber data. Sumber data yang akan diekstrak berasal dari berbagai macam sumber data dengan berbagai *Database* Management System, *Operating System* dan protokol yang digunakan. Oleh karena itu, dalam proses ektraks data harus dilakukan secara efektif.

b. *Transformation*

   Pada tahap ini, dilakukan proses *cleaning* dan *conforming* agar data tersebut menjadi akurat sehingga data tersebut tepat, lengkap, konsisten, dan jelas. *Transformation* memiliki proses yaitu data *cleaning*, *transformation*, dan *integration*. Dalam tahapan ini, didefinisikan granularity dari tabel fakta, tabel dimensi, dan skema *data warehouse* (Star Schema atau Snowflake). Table fakta adalah pusat dari skema *data warehouse* yang umumnya mengandung measure yang merupakan salah satu property yang berisikan perhitungan untuk mengukur tingkat analisis. Tabel Dimensi adalah tabel yang berisikan data detail yang berhubungan dengan tabel fakta. Skema *data warehouse* adalah suatu skema yang menghubungkan tabel fakta dan tabel dimensi.

c. *Loading*

   Pemuatan data ke target struktur yang multi dimensi adalah tahapan akhir dalam ETL. Dalam tahap ini, proses *Extraction* dan *Transformation* disajikan ke dalam struktur dimensi yang dapat diakses oleh user dalam system aplikasi. Tahapan loading memiliki proses Loading Dimension dan Loading Fact.



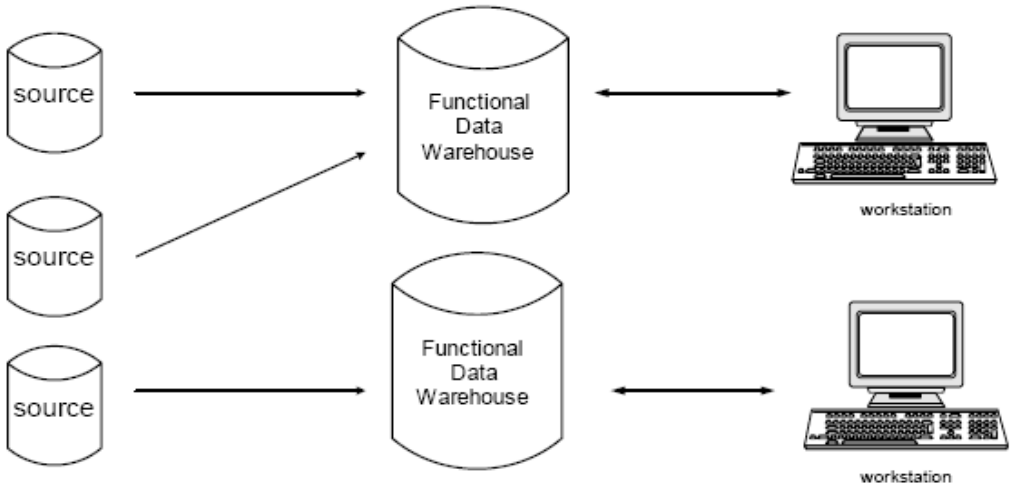
Gambar 2.8 *Framework* ETL [4]

2.3.5.5 Jenis Dasar *Data warehouse*

Berikut ini adalah tiga jenis dasar sistem *Data warehouse* :

1. *Functional Data warehouse* (Data warehouse Fungsional)

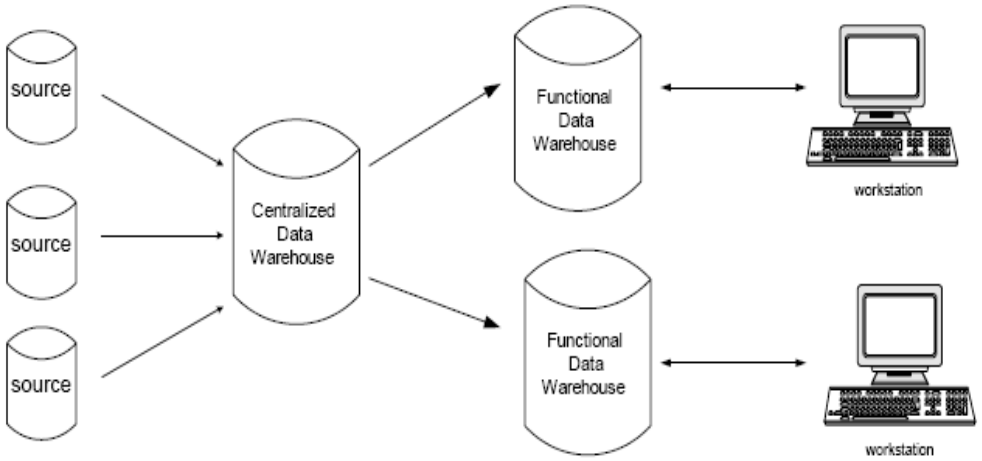
   Kata operasional disini merupakan *database* yang diperoleh dari kegiatan sehari-hari. *Data warehouse* dibuat lebih dari satu dan dikelompokkan berdasar fungsi-fungsi yang ada di dalam perusahaan seperti fungsi keuangan (financial), marketing, personalia dan lain-lain. Keuntungan dari bentuk *data warehouse* seperti ini adalah, sistem mudah dibangun dengan biaya relatif murah sedangkan kerugiannya adalah resiko kehilangan konsistensi data dan terbatasnya kemampuan dalam pengumpulan data bagi pengguna.



Gambar 2.9 Bentuk *data warehouse fungsional* [5]

2. *Centralized Data warehouse* (Data warehouse Terpusat)

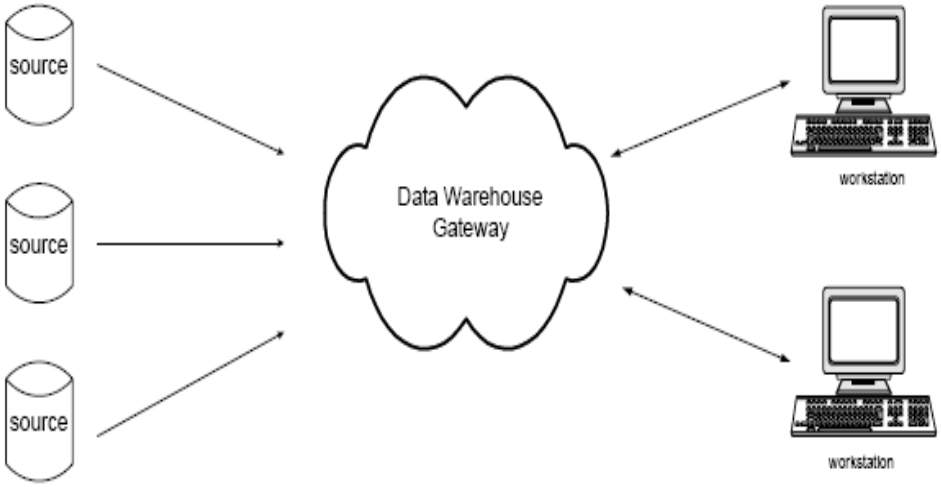
   Bentuk ini terlihat seperti bentuk *data warehouse* fungsional, namun terlebih dahulu sumber data dikumpulkan dalam satu tempat terpusat, kemudian data disebar ke dalam fungsinya masing-masing, sesuai kebutuhan persuhaan. *Data warehouse* terpusat ini, biasa digunakan oleh perusahaan yang belum memiliki jaringan eksternal. Keuntungan dari bentuk ini adalah data benar - benar terpadu karena konsistensinya yang tinggi sedang kerugiannya adalah biaya yang mahal serta memerlukan waktu yang cukup lama untuk membangunnya.



Gambar 2.10 Bentuk *data warehouse* terpusat [5]

3. *Distributed Data warehouse* (*Data warehouse* terdistribusi)

   Pada *data warehouse* terdistribusi ini, digunakan gateway yang berfungsi sebagai jembatan penghubung antara *data warehouse* dengan workstation yang menggunakan sistem beraneka ragam. Dengan sistem terdistribusi seperti ini memungkinkan perusahaan dapat mengakses sumber data yang berada diluar lokasi perusahaan(*eksternal*). Keuntungannya adalah data tetap konsisten karena sebelum data digunakan data terlebih dahulu di sesuaikan atau mengalami proses sinkronisasi. Sedangkan kerugiannya adalah lebih kompleks untuk diterapkan karena sistem operasi dikelola secara terpisah juga biaya nya yang paling mahal dibandingkan dengan dua bentuk *data warehouse* lainnya.



Gambar 2.11 Bentuk *Data Warehouse* Terdistribusi [5]

2.3.5.6 Metodologi Perancangan *Database* untuk *Data warehouse*

Menurut Connoly, metodologi *data warehouse* yaitu *step – nine methodolgy*. Dimana terdiri dari beberapa step yaitu [2]:

1. Memilih Proses

   Fungsi dari pemilihan proses mengacu pada sebuah permasalahan subjek yang merupakan bagian dari *data mart*. Data mart yang sering digunakan adalah data yang sering digunakan untuk memenuhi sebuah kebutuhan dan menjawab sebuah permasalahan bisnis.
2. Memilih Grain

   Memutuskan dengan tepat apa yang akan ditampilkan di tabel fakta. Dengan memilih grain kita dapat menentukan atau mengidentifikasi dimensi-dimensi apa saja yang terdapat di tabel fakta.
3. Identifikasi dan membuat dimensi yang sesuai

   Sebuah dimensi yang baik dapat membuat data mart mudah di mengerti dan digunakan.
4. Memilih Fakta

   Grain dari tabel fakta dapat menentukan data apa saja yang di gunakan untuk data mart.

5. Menentukan pra-kalkulasi dalam tabel fakta

   Setiap fakta-fakta yang sudah terpilih harus diuji ulang terlebih dahulu untuk menentukan apakah terdapat peluang untuk menentukan pra-kalkulasi.

6. Melengkapi tabel dimensi

   Pada tahap ini kita melengkapi sebanyak mungkin untuk mendeskripsikan tabel dimensi tersebut yang memiliki hubungan dengan tabel dimensi itu sendiri.

7. Memilih durasi dari *database*

   Durasi menunjukan waktu yang digunakan untuk menjalankan sebuah tabel fakta. Pada tahap ini yang dilakukan adalah menentukan batas waktu dari data yang akan di masukan kedalam *data warehouse*

8. Melacak perubahan dari dimenesi secara acak

   Dalam melacak perubahan dimensi, ada 3 tipe yang dapat digunakan. Antara lain :

   Tipe 1 : Perubahan atribut pada sebuah dimensi akan menyebabkan atribut tersebut akan terganti (*overwrite*).

   Tipe 2 : Perubahan atribut pada sebuah dimensi akan menyebabkan dimasukannya data baru kedalam dimensi tersebut.

   Tipe 3 : Atribut pada sebuah dimensi akan menyebabkan penambahan atribut alternatif, sehingga data yang lama dengan data yang baru dapat diakses secara bersamaan dengan dimensi yang sama.

9. Memutuskan prioritas dan cara query

   Pada tahap ini sudah mempertimbangkan perancangan fisikal yang mempengaruhi persepsi pengguna dalam menggunakan *data mart*. Yaitu urutan dari tabel fakta secara fisik dan ke munculan ringkasan.

2.3.5.7 Konsep pemodelan *Data warehouse*

Menurut Connolly, permodelan dimensional menggunakan konsep pemodelan *Entity-Relationship* (ER) dengan beberapa batasan - batasan penting. Setiap model dimensional tersusun dari satu table dengan sebuat composite

primary key, dinamakan tabel fakta, dan satu set tabel - tabel yang lebih kecil yang bernama tabel dimensi. Setiap table dimensi mempunyai sebuah primary key sederhana (*non-composite*) yang berhubungan dengan satu komponen dari *composite key* di tabel fakta. Dengan kata lain, *primary key* dari tabel fakta dibuat dari dua atau lebih *foreign key* [2].

1. Tabel Fakta

   Menurut Ralph Kimbal, Margy Ross tabel fakta adalah tabel utama dalam model dimensi dimana numerik pengukuran kinerja bisnis yang disimpan [5].



Gambar 2.12 Contoh Tabel Fakta [6]

   Tabel fakta pada umumnya memiliki sebuah *primary key*, dan biasanya disebut *composite* atau *concatenated key*. Setiap tabel dalam model dimensi memiliki *composite key*, dan tabel yang memiliki *composite key* adalah tabel fakta. Dan setiap tabel yang memiliki hubungan *many to many* (banyak –ke -banyak) harus menjadi tabel fakta dan yang lainnya menjadi tabel dimensi.

2. Tabel Dimensi

   Menurut Ralph Kimbal, Margy Ross tabel dimensi adalah sebuah tabel yang memiliki banyak kolom atau atribut. Atribut ini menggambarkan baris dalam tabel dimensi, dan setiap dimensi didefinisikan oleh satu *primary key*. Ditunjuk oleh notasi PK, yang berfungsi sebagai dasar untuk penghubung antara tabel dimensi dengan tabel fakta [5].

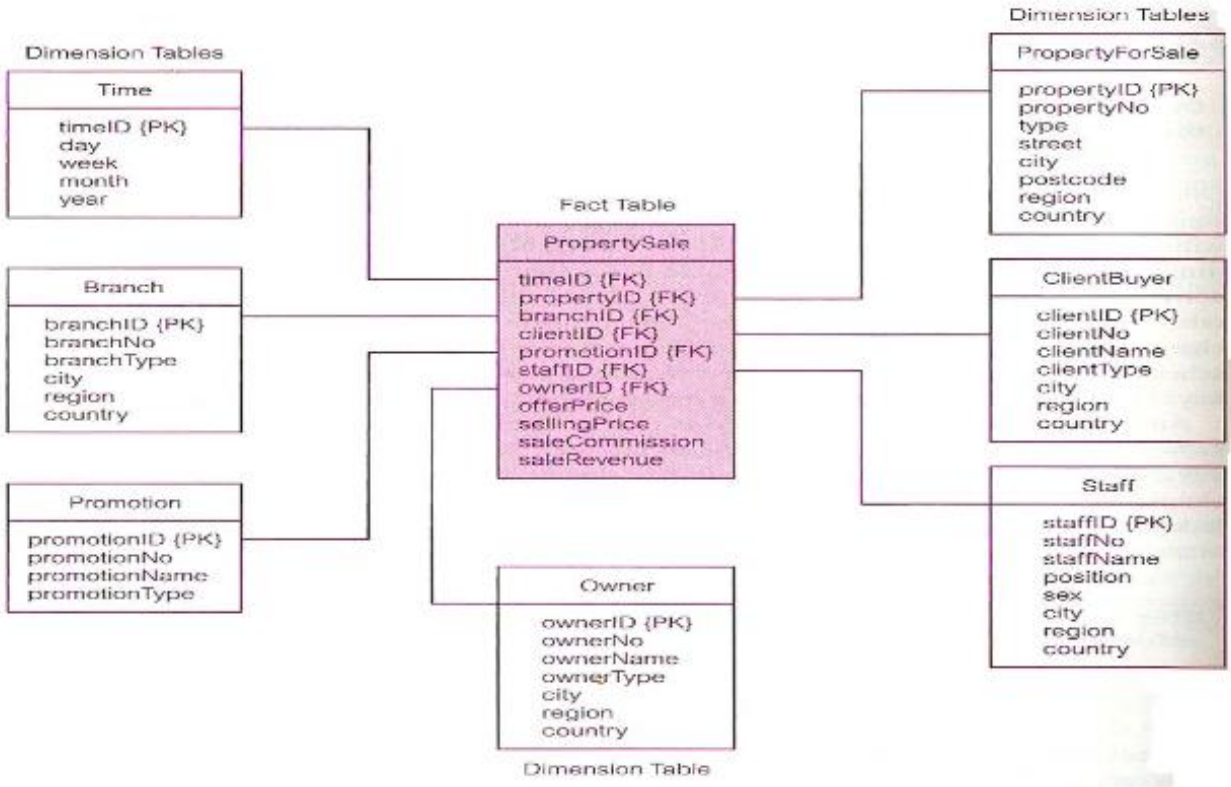


Gambar 2.13 Contoh Tabel Dimensi [6]

3. Skema Bintang

   Menurut Thomas Connolly dan Carolyn Begg, skema bintang adalah model dimensional data yang memiliki *fact table* di tengah, dikelilingi oleh *denormalized dimension tables* [2]. Selain itu skema bintang memudahkan *end – user* untuk memahami struktur *database* pada *data warehouse* yang di rancang. Keuntungan dari penggunaan skema bintang :

   1. Respon data lebih cepat daripada perancangan *database* operasional.
   2. Mempermudah dalam hal modifikasi atau pengembangan *data warehouse* yang terus menerus.
   3. *End-user* dapat menyesuaikan cara berpikir dan menggunakan data.
   4. Menyederhanakan pemahaman dan penulusuran metadata bagi pemakai dan pengembang.

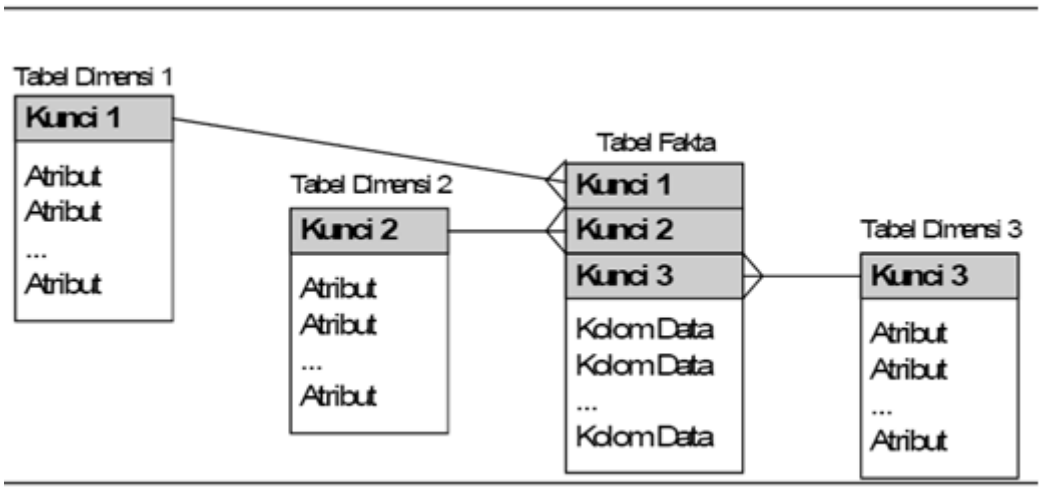


Gambar 2.14 skema bintang [2]

Beberapa jenis skema bintang, antara lain:

a. Skema bintang sederhana

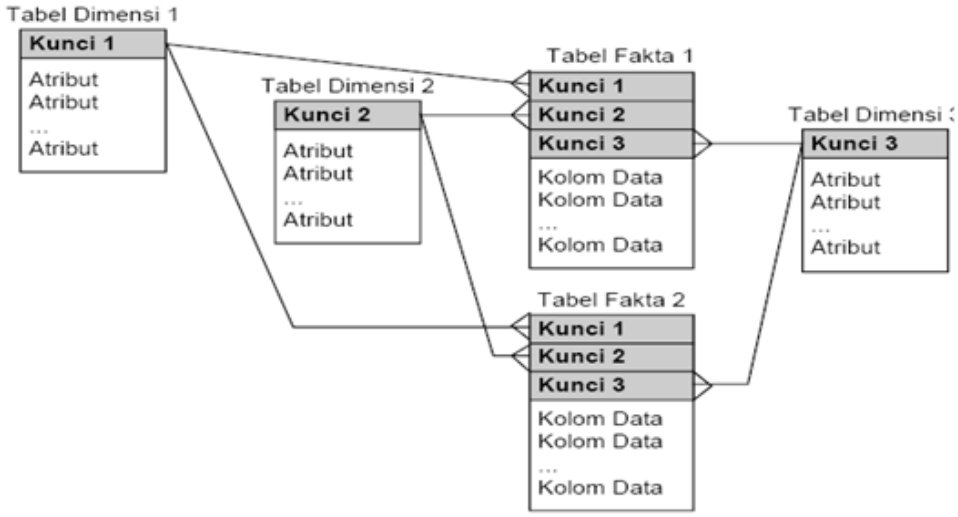
Dalam skema ini, setiap table harus memiliki *primary key* yang terdiri dari satu kolom atau lebih. *Primary key* dari table fakta terdiri dari satu atau lebih *foreign key. Foreign key* merupakan *primary key* pada table lain.



Gambar 2.15 Skema Bintang Sederhana [6]

b. Skema bintang dengan banyak table fakta

Skema bintang juga bisa terdiri dari satu atau lebih table fakta. Dikarenakan karena table fakta tersebut ada banyak, misalnya disamping penjualan terdapat table fakta *forecasting* dan *result*. Walaupun terdapat lebih dari satu table fakta, mereka tetap menggunakan table dimensi bersama-sama.

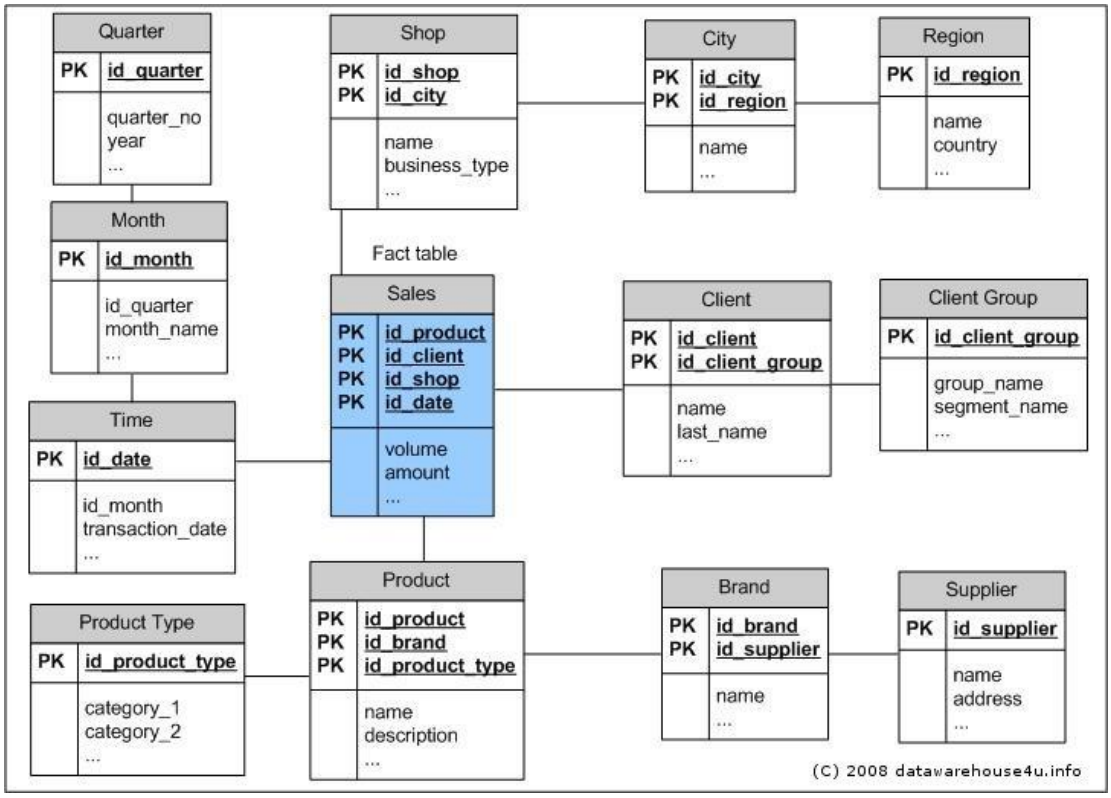


Gambar 2.16 Skema Bintang Dengan Banyak Tabel Fakta [6]

4. *Snowflake Schema*

   Menurut Connolly skema snowflake adalah bentuk lain dari skema bintang dimana tabel dimensi tidak mengandung data yang telah didenormalisasi [2]. Keuntungan *Snowflake Schema*, antara lain:

   a. Kecepatan memindahkan data dari data OLTP ke dalam *Metadata*.
   b. Sebagai kebutuhan dari alat pengambil keputusan tingkat tinggi dimana dngan tipe yang seperti ini seluruh struktur dapat digunakan sepenuhnya.
   c. Banyak yang beranggapan lebih nyaman merancang dalam bentuk normal ketiga.

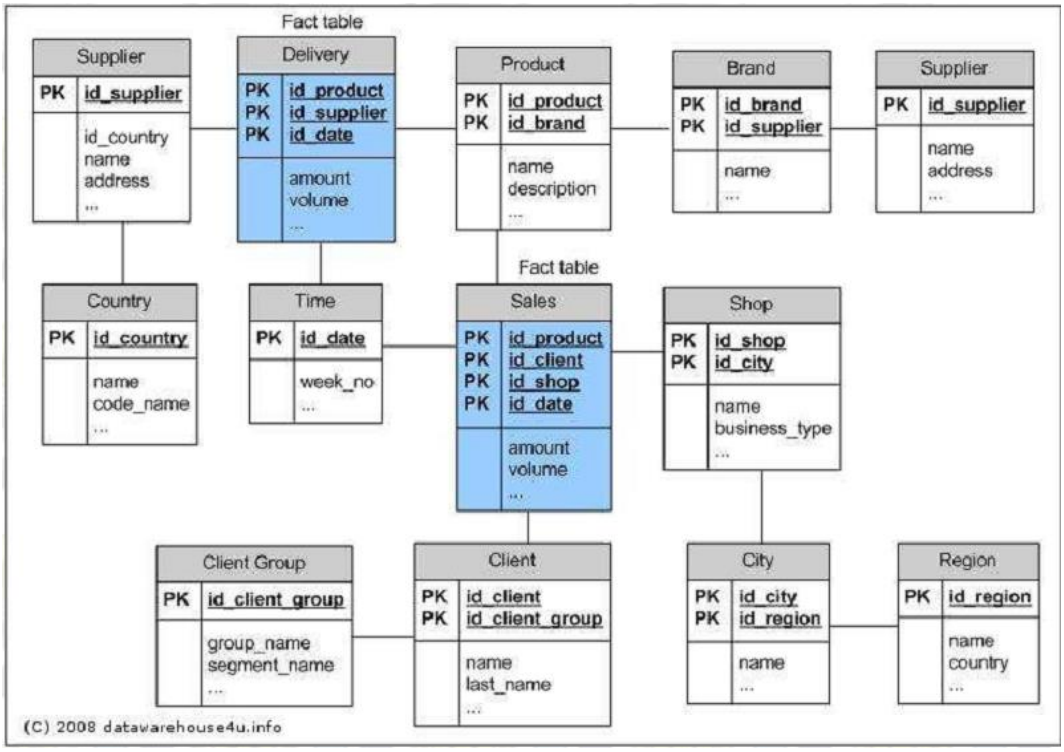


Gambar 2.17 *snowflake schema* [7]

5. *Fact constellation schema*

   Fact constellation schema adalah dimensional model yang didalamnya terdapat lebih dari satu tabel fakta yang membagi satu atau lebih dimension table. Skema ini lebih kompleks daripada star skema karena

berisi berbagai tabel fakta. Dalam *fact constellation schema*, satu dimensi table bisa digunakan di beberapa tabel fakta sehingga desainnya lebih kompleks. Keuntungan dari *fact constellation schema* adalah kemampuan untuk memodelkan bisnis lebih akurat menggunakan beberapa tabel fakta. Namun kerugiannya adalah sulit dalam pengelolaan dan desain yang rumit



Gambar 2.18 *Fact Constellation Schema* [7]

2.3.5.8 Tujuan *Data warehouse*

Menurut Connolly dan Begg, tujuan *data warehouse* adalah mengintegrasikan data organisasi dengan jumlah yang besar menjadi satu wadah dimana pengguna dapat dengan mudah menjalankan *query*, membuat laporan serta melakukan analisis. Singkatnya, *data warehouse* adalah teknologi manajemen dan analisis data [2].

2.3.5.9 Keuntungan *Data warehouse*

Menurut Connolly dan Begg keberhasilan implementasi dari sebuah *data warehouse* dapat membawa manfaat yang signifikan untuk sebuah organisasi seperti di antaranya [2] :

a. Pengembalian

   investasi yang berpotensi tinggi. Sebuah organisasi harus berkomitmen dalam berinvestasi pada banyak sumber daya untuk menjamin keberhasilan penerapan *data warehouse* dan biaya yang diinvestasikan dapat sangat bervariasi mulai dari £ 50.000 sampai lebih dari 10 juta pound karena berbagai solusi teknis yang tersedia. Namun, sebuah studi oleh *International Data Corporation* (IDC) pada tahun 1996 melaporkan bahwa dalam tiga tahun biaya yang kembali rata-rata atas investasi (ROI) dalam *data warehouse* mencapai 401%, dengan lebih dari 90% dari perusahaan yang disurvei mencapai lebih dari 40% ROI, setengah perusahaan mencapai lebih dari 160% ROI, dan seperempat dengan lebih dari 600% ROI (IDC, 1996).

b. Keunggulan kompetitif. Pengembalikan investasi yang besar bagi perusahaan yang telah berhasil menerapkan *data warehouse* adalah bukti dari keunggulan kompetitif yang sangat besar yang datang dengan teknologi ini. Keunggulan kompetitif yang diperoleh dengan membiarkan akses ke penetapan *data marker* dapat mengungkapkan informasi yang tersedia sebelumnya, tidak dikenal dan belum dimanfaatkan, misalnya pelanggan, trend dan tuntutan.

c. Peningkatan

   produktivitas dalam pengambilan keputusan perusahaan. *Data warehouse* dapat meningkatkan produktivitas para pengambil keputusan perusahaan dengan menciptakan *database* yang terintegrasi, konsisten dan berfokus pada subjek dan data historis. Mengintegrasikan data dari beberapa sistem yang tidak kompatibel ke dalam bentuk yang menyediakan tampilan konsisten organisasi. Dengan mengubah data menjadi informasi

yang bermakna, *data warehouse* memungkinkan para top management perusahaan untuk menganalisis dengan lebih substantif, akurat dan konsisten

2.3.5.10 *Data Warehouse Tools*

Berikut ini adalah tools yang digunakan user setelah data warehouse terbentuk dengan tujuan yang berbeda-beda[13]:

1. OLAP(On-Line Analytical Processing) OLAP merupakan salah satu data warehouse tools untuk melakukan analisis data. OLAP sendiri adalah suatu teknologi yang dirancang untuk memberikan kinerja yang unggul untuk ad hoc business intelligence queries[13]. OLAP dirancang untuk beroperasi secara efisien dengan data yang terorganisir sesuai dengan model dimensi umum yang biasa digunakan dalam data warehouse. Tidak ada data warehouse modern yang selesai dengan sempurna tanpa fungsionalitas OLAP. Tanpa OLAP, kita tidak dapat memberikan users seluruh kemampuan untuk melakukan analisis multidimensional, untuk melihat informasi dari segala sudut pandang, dan untuk membuat keputusan yang bersifat kritikal. Oleh karena itu, OLAP sangat krusial.[14]. Berikut ini adalah karakteristik paling mendasar dalam OLAP systems[14]:
   a) Memberikan para pelaku bisnis pandangan logis yang multidimensi dari data yang ada dalam data warehouse.
   b) Memfasilitasi query interaktif dan kompleks analisis kepada users.
   c) Memungkinkan user untuk drill-down atau roll-up data yang ada baik untuk single dimension maupun untuk multi dimension.
   d) Memberikan kemampuan untuk melakukan perhitungan danmperbandinganmyang rumit.
   e) Hasilnya bias dipresentasikan dalam bentuk yang lebih berarti, seperti grafik atau tabel.

   Kegunaan OLAP[14]antara lain:

   a) Meningkatkan produktivitas dari bisnis manajer, eksekutif dan analis.

b) Memanfaatkan OLAP dengan baik bisa membuat users bisa dengan percaya diri membuat analisis mereka sendiri tanpa bantuan IT assistance.

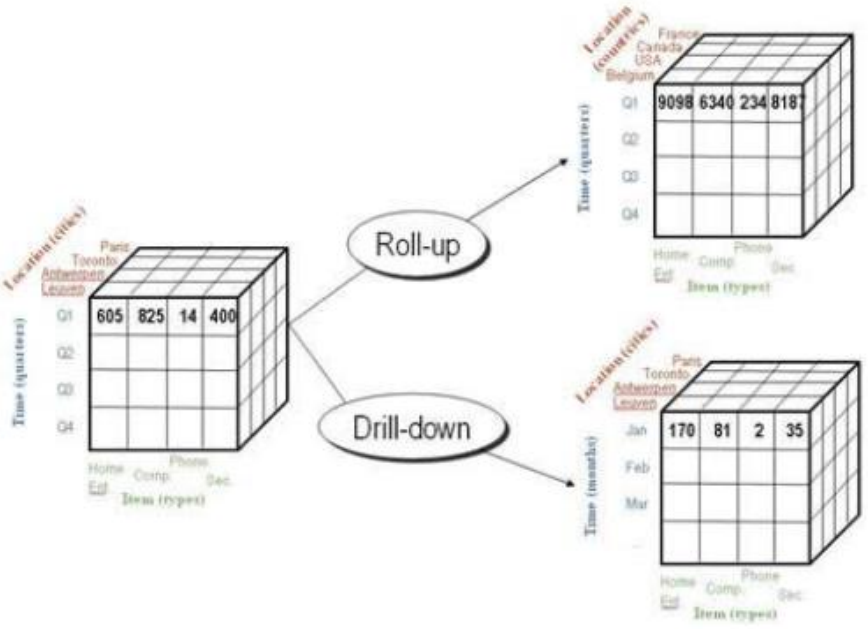
c) Keuntungan untuk IT developers yaitu penggunaan OLAP bias sangat membantu mempercepat kinerja dari aplikasinya sendiri.

d) Meningkatkan efisiensi kerja.OLAP dapat digunakan untuk melakukan hal-hal seperti[15]:

   a) Consolidation (roll-up) Konsolidasi melibatkan pengelompokkan data.

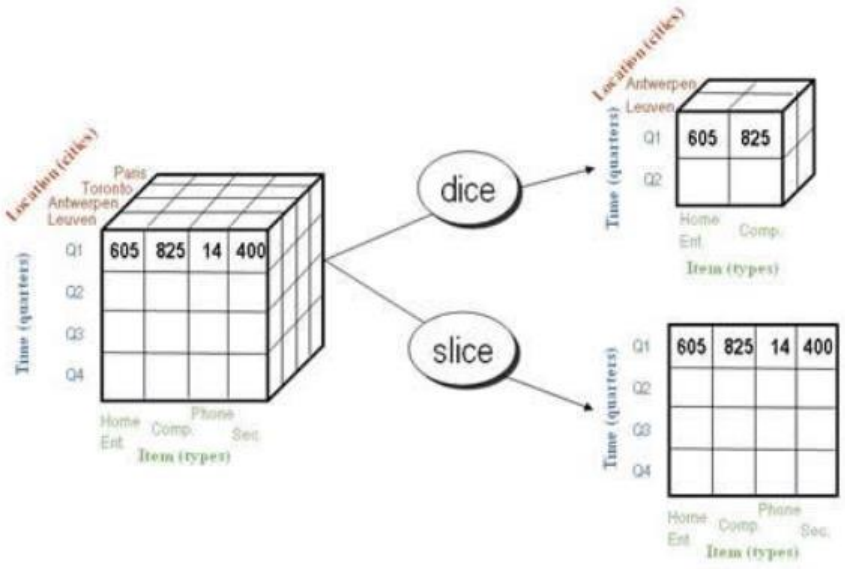
   b) Drill-down Suatu bentuk yang merupakan kebalikan dari konsolidasi, untuk mendapatkan lebih detail tentang suatu dimensi serta bisa dikatakan sebagai suatu navigasi dari tingkat yang lebih umum ke tingkat yang lebih spesifik. Untuk lebih jelasnya roll-up dan drill-down dapat dilihat pada Gambar 2.19

.



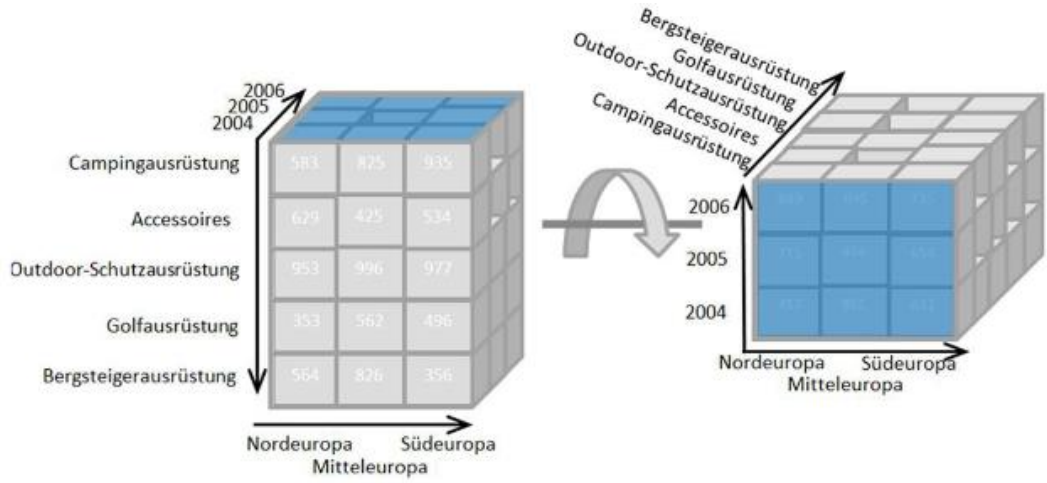
Gambar 2.19 *roll-up* dan *drill-down*

c) Slicing and dicing Menjabarkan pada kemampuan untuk melihat data dari berbagai sudut pandang. Untuk lebih jelasnya slicing and dicing bisa dilihat pada Gambar 2.20.



Gambar 2.20 *slicing* dan *dicing*

d) PivotMenampilkan nilai-nilai ukuran dalam tata letak tabel yang berbeda dan juga bisa mengatur kembali dimensi dalam OLAP cube. Untuk lebih jelasnya pivot bisa dilihat pada Gambar 2.21.



Gambar 2.21 Pivot

Untuk memodelkan OLAP kepada users, kita gunakan OLAP cube. OLAP cube adalah bagian utama dari OLAP yang berisi kumpulan data yang banyak dan disatukan (agregasi) untuk mempercepat hasil query[15].

OLAP cube contohnya seperti pada Gambar 2.19, Gambar 2.20, dan Gambar 2.21.

2. Reporting

   Reporting tools merupakan tools yang digunakan untuk mempermudah user memperoleh data yang sudah lama ataupun data sekarang dan melakukan beberapa standard analisis statistik[14]. Data yang dihasilkan dari reporting tools bisa berupa bentuk laporan biasa dan juga bisa berupa grafik.

3. Data mining Data mining merupakan teknologi yang mengaplikasikan algoritma yang canggih dan kompleks untuk menganalisis data dan mencari informasi yang menarik dari kumpulan data tersebut. Perbedaan mendasar antara OLAP dan data mining yaitu terletak pada apa yang akan dianlisisnya. Pada OLAP, yang dianalisis adalah modelnya, tetapi pada data mining yang dianalisis adalah datanya (harus berjumlah besar).[13].

2.3.6 The Systems Development Life Cycle (SDLC)

Menurut Satzinger, *Systems Development Life Cycle* (SDLC) adalah seluruh proses yang membangun, menyebarkan, menggunakan, dan memperbarui sistem informasi [8]. *Sytems Development Life Cycle* (SDLC) merupakan pengembangan system dari pertama kali dipelajari dan digunakan hingga system tersebut mengalami pembaharuan, peningkatan atau tergantikan dengan sistem yang lebih baik.

2.3.6.1 Tahapan dalam Systems Development Life Cycle (SDLC)

Dalam *Systems Development Life Cycle* (SDLC) terdapat beberapa tahapan dalam perancangannya yaitu :

Tahap 1 : *Project Planning Phase*

Tahapan ini untuk mengidentifikasi cakupan dari sistem baru, dan memastikan bahwa project tersebut layak menggunakan sistem baru, sumber-sumber data untuk project tersebut dan batasan budget nya.

Tahap 2: *Analysis Phase*

Di Tahapan ini akan dilakukan pemahaman terutama terhadap dokumen kebutuhan bisnis secara rinci dan juga persyaratan pengelolaan sistem yang baru.

Tahap 3 : *Design Phase*

Untuk merancang sistem solusi berdasarkan persyaratan yang ditetapkan dan keputusan yang dibuat selama analisis.

Tahap 4: *Implementation Phase*

Untuk membangun tes dan memasang sebuah sistem yang dapat dipercaya dilengkapi dengan user yang terlatih dan siap untuk mendapatkan keuntungan seperti yang diharapkan sebelumnya dari penggunaan sistem tersebut.

Tahap 5 : *Support Phase*

Untuk menjaga sistem agar berjalan secara produktif dari awal dibangun sistem tersebut hinga bertahun-tahun sampai di mana masa hidup sistem tersebut berakhir.

2.3.7 OOAD (*Object Oriented Analysis Design*)

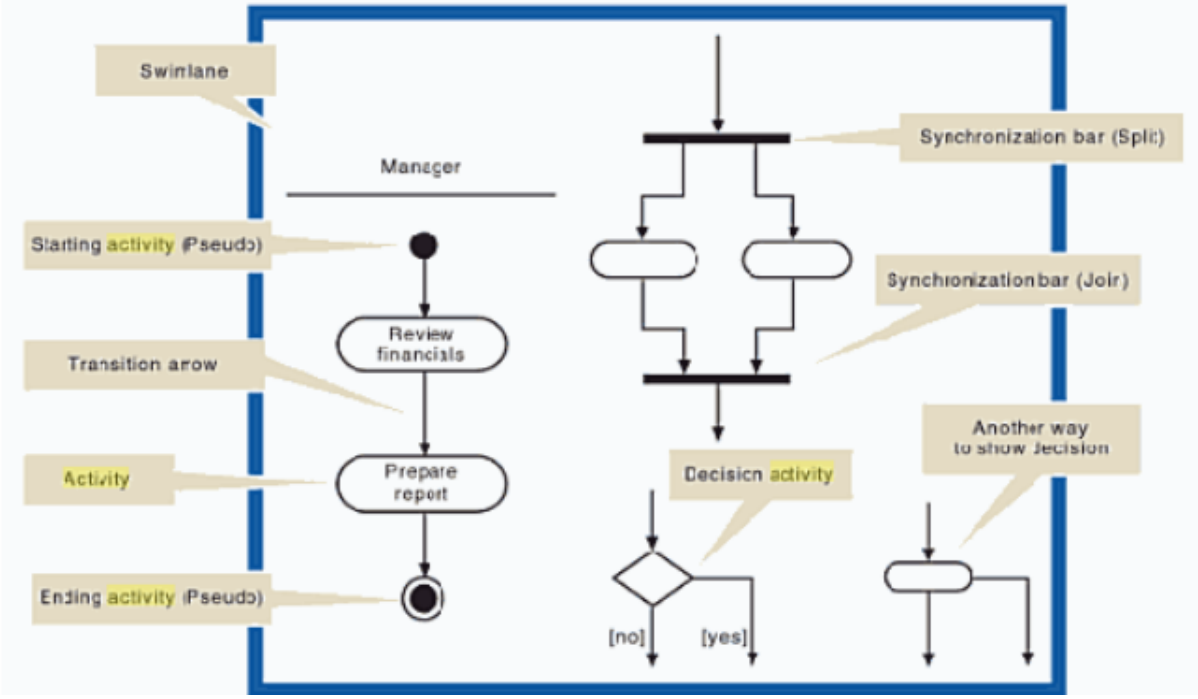
Menurut Hanif Al Fatta *Object Oriented Analysis Design* merupakan teknik yang mengintegrasikan data dan proses yang disebut objek *Object-Oriented Analysis* (OOA) adalah semua jenis objek yang melakukan pekerjaan dalam sistem dan menunjukkan interaksi pengguna apa yang dibutuhkan untuk menyelesaikan tugas tersebut [7]. Object diartikan suatu hal dalam system computer yang dapat merespon pesan [8]. *Object-Oriented Design* (OOD) adalah semua jenis objek yang diperlukan untuk berkomunikasi dengan orang dan perangkat dalam sistem, menunjukkan bagaimana objek berinteraksi untuk menyelesaikan tugas, dan menyempurnakan definisi dari masing-masing jenis objek sehingga dapat diimplementasikan dengan bahasa tertentu atau lingkungan [8]. *Object-Oriented Programming* (OOP) menuliskan laporan dalam bahasa pemrograman untuk mendefinisikan apa yang setiap jenis objek ini termasuk pesan bahwa pengirim satu sama lain [8].

2.3.7.1 Unified Modeling Language (UML)

Menurut Satzinger, *Unified Modeling Language* (UML) merupakan suatu set standar konstruksi model dan notasi dikembangkan secara khusus untuk pengembangan berorientasi objek *Unified Modelling Language* (UML) merupakan suatu bahasa pemodelan untuk membuat, mendokumentasikan, menggambarkan sistem informasi [8].

2.3.7.2 Activity Diagram

Menurut Satzinger, sebuah *activity diagram* hanyalah sebuah diagram alur kerja yang menggambarkan berbagai pengguna kegiatan, orang yang melakukan aktivitas masing-masing, dan aliran sekuensial kegiatan ini atau dapat dikatakan diagram yang menggambarkan alur proses bisnis [8].

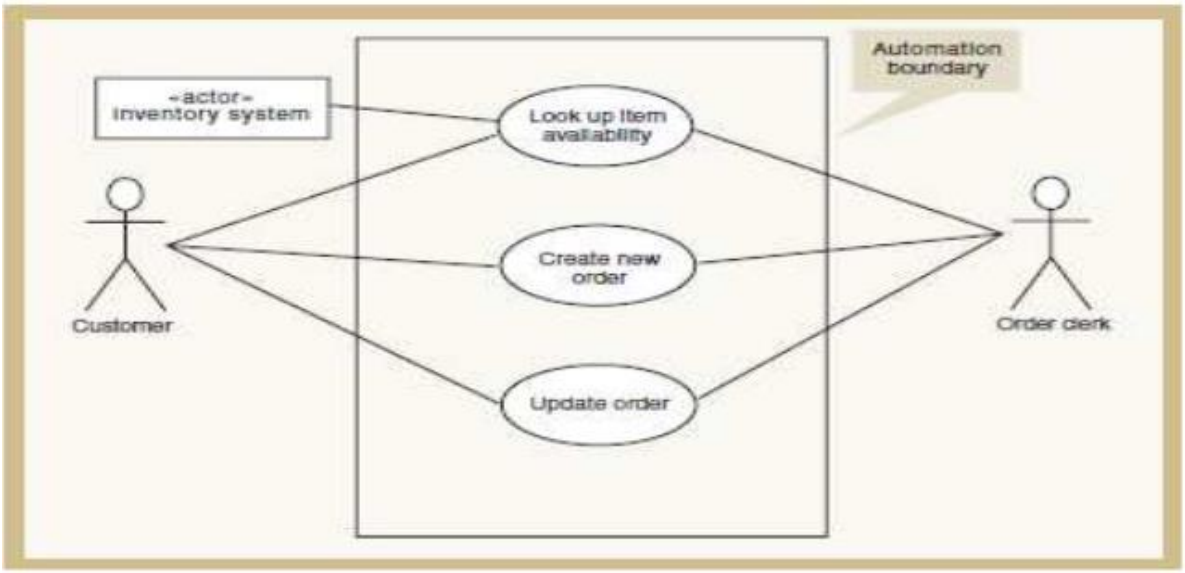


Gambar 2.22 *Activity Diagram* [9]

2.3.7.3 Use Case

Use Case merupakan model fungsional yang di dalamnya terdapat *actor* dan *use case* itu sendiri, actor disini merupakan orang-orang yang menjalankan

aktifitasnya dan berhubungan langsung dengan sistem, sedangkan *use case* itu sendiri berisi pekerjaan yang dilakukan actor yang menggunakan sistem. Berikut merupakan symbol yang digunakan dalam *Use Case Diagram* :



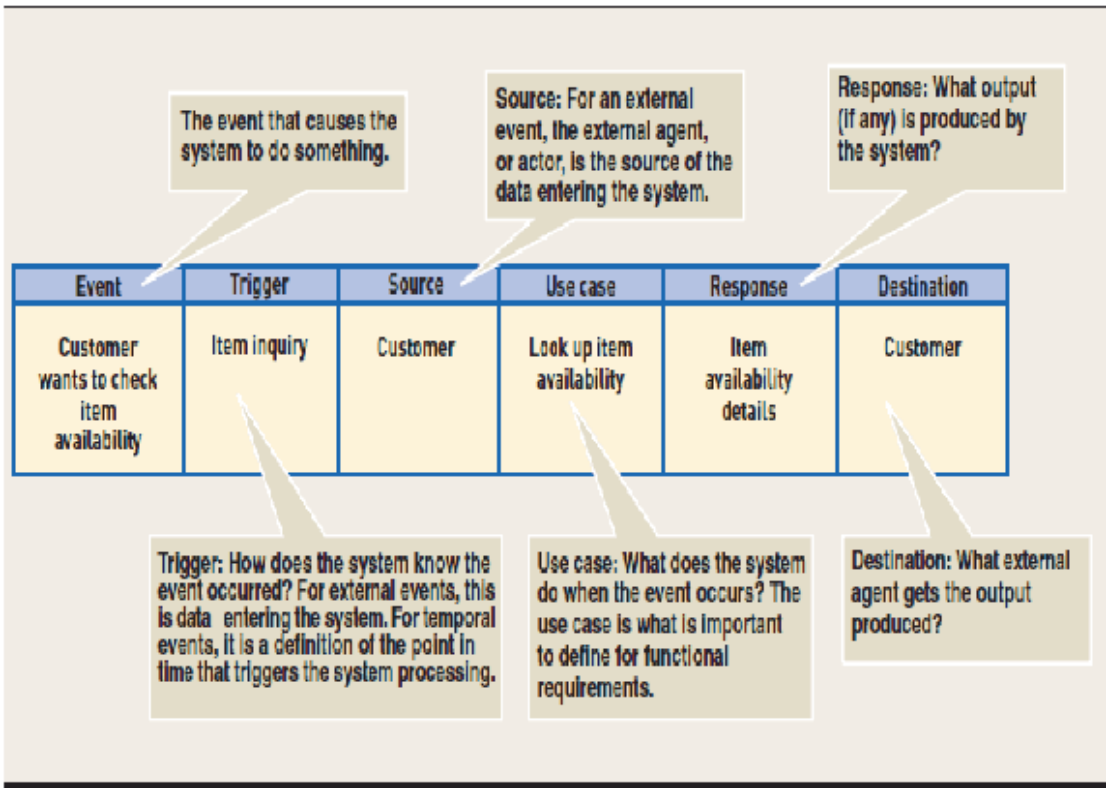
Gambar 2.23 *Use Case diagram* [9]

2.3.7.4 Event Table

Event table adalah sebuah table yang meliputi baris dan kolom, yang berisi beberapa komponen yang mewakili peristiwa dan rincian dari peristiwa mereka, masing-masing. Berikut penjelasan komponen dari *event table* antara lain:

a. *Event* Merupakan katalog use case daftar peristiwa dalam baris dan potongan kunci informasi tentang setiap peristiwa dalam kolom.
b. *Trigger* Merupakan sinyal yang memberitahukan system bahwa suatu peristiwa telah terjadi, baik kedatangan membutuhkan pengolahan data atau titik waktu.
c. *Source* Merupakan agen eksternal atau actor yang memasok data ke sistem.
d. *Response* Merupakan output yang dihasilkan oleh sistem, yang menuju ke tujuan.

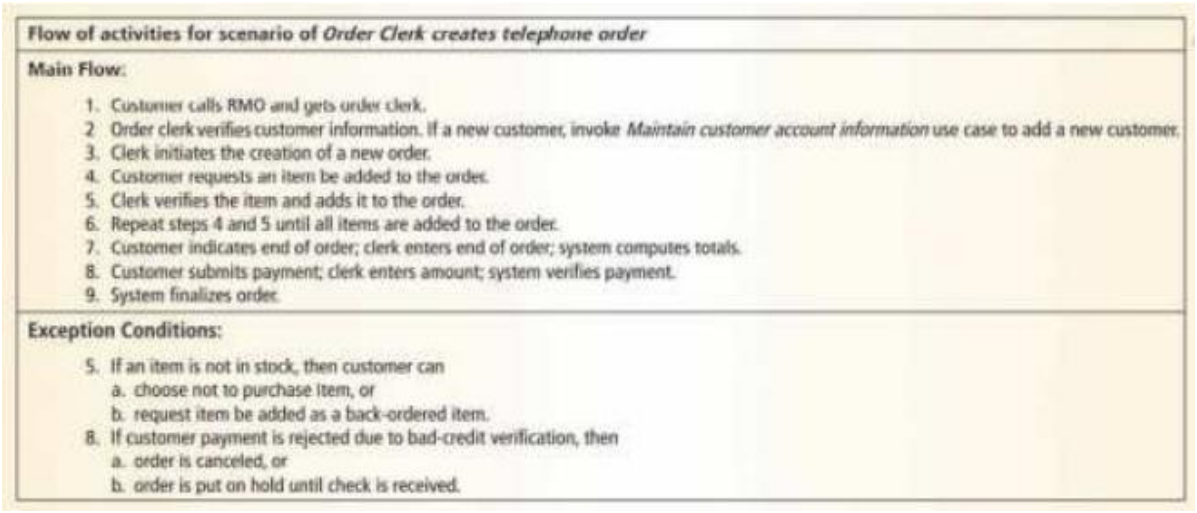
e. *Destination* Merupakan agen eksternal atau *actor* yang menerima data dari sistem.



Gambar 2.24 *Event Table* [9]

2.3.7.5 Use Case Description

Deskripsi atau penjelasan mengenai *Use Case* yang berisi nama *use casenya*, *main flow*. Di sini dijelaskan tentang bagaimana actor yang menggunakan system yang telah digambarkan di *Use Case*.



Flow of activities for scenario of *Order Clerk creates telephone order*

Main Flow:

1. Customer calls RMO and gets order clerk.
2. Order clerk verifies customer information. If a new customer, invoke *Maintain customer account information* use case to add a new customer.
3. Clerk initiates the creation of a new order.
4. Customer requests an item be added to the order.
5. Clerk verifies the item and adds it to the order.
6. Repeat steps 4 and 5 until all items are added to the order.
7. Customer indicates end of order; clerk enters end of order; system computes totals.
8. Customer submits payment; clerk enters amount; system verifies payment.
9. System finalizes order.

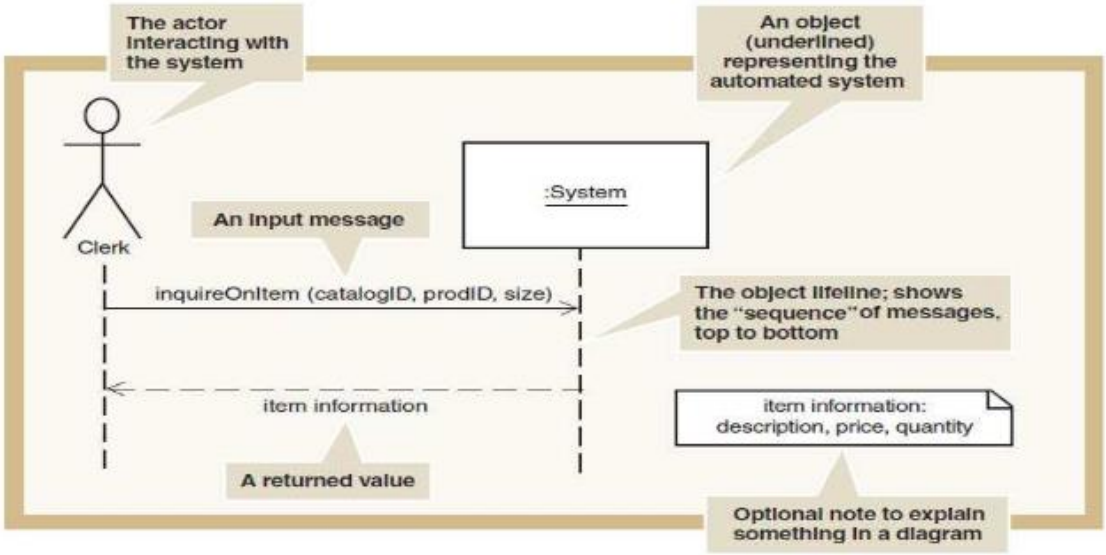
Exception Conditions:

5. If an item is not in stock, then customer can
   a. choose not to purchase item, or
   b. request item be added as a back-ordered item.
8. If customer payment is rejected due to bad-credit verification, then
   a. order is canceled, or
   b. order is put on hold until check is received.

Gambar 2.25 *Use Case Description* [9]

2.3.7.6 Sequence Diagram

Sequence Diagram biasanya digunakan dalam hubungannya dengan deskripsi menggunakan kasus untuk membantu dokumen rincian kasus penggunaan tunggal atau skenario dalam kasus penggunaan atau menangkap interaksi antara system dan *eksternal entity* yang direpresentasikan oleh *actor*.



Gambar 2.26 *sequence diagram* [9]

2.3.8 C#

Bahasa pemrograman C# dikembangkan oleh *Microsoft* sebagai bahasa yang simple, modern, *general purpose*, dan berorientasi objek. Pengembangan bahasa C# sangat dipengaruhi oleh bahasa pemrograman terdahulu, terutama C++. Delphi, dan Java. C++ dikenal memiliki kecepatan yang tinggi dan memiliki akses memori hapir hingga ke *low level*. Namun bagi para programmer, C++ merupakan Bahasa yang relatif rumit dibandingkan bahasa pemrograman lainnya. Kehadiran C# memberi suntikan optimisme bagi para programmer untuk dapat mengembangkan aplikasi yang berdasa guna dengan baik cepat dan lebih mudah. Bahasa C# masih harus dikembangkan dengan kemampuan untuk melakukan berbagai *task*. Namun dalam hal performansi hingga saat ini C++ masih diakui sebagai salah satu yang terbaik.

C# dikembangkan sejalan dengan pengembangan teknologi .Net. Teknologi .NET telah berevolusi dari .NET 1.0 hingga .NET 4.0. Sendiri merupakan sebuah *framework* yang memiliki *base class library*, dan bisa diimplementasikan ke dalam beberpaa bahasa pemrograman yang dikembangkan *Microsoft* termasuk diantaranya adalah bahasa C#. *Microsoft* mengembangkan IDE yang secara khusus mendukung pengembangan aplikasi dengan teknologi .NET, versi terakhirnya yaitu Visual Studio 2010, mendukung teknologi .Net 4. Diantara karakteristik dari .NET adanya sebuah *Common Langguage Runtime* (CLR). CLR sendiri merupakan sebuah komponen *virtual machine* yang akan diakses program pada saat *runtime* kode yang ditulis dalam bahasa C# ataupun VB.NET misalnya *dicompile* oleh .NET *Compiler* menjadi code dalam format *Common Intermediate Langguage* (CIL).

CIL merupakan sebuah format bahasa standar pada *level intermediate* yang digunakan dalam bidang teknologi .NET apa pun bahasa pemrograman yang digunakan oleh kode. Kemudian pada saat *runtime*, CLR mengeksekusi CIL dengan melakukan proses loading dan linkin serta menghasilkan bahasa mesin untuk mengeksekusi program. Proses ini memunculkan istilah *multilangguage* pada pengembangan aplikasi .Net programmer memiliki pilihan untuk

menentukan pilihan untuk menggunakan bahasa pemrograman, yang familiar bagi mereka dan apapun bahasa pemgraman apapun mereka tetap memilih akan tetap memiliki keungulan-keungulan dan teknologi lainnya.

Standar *European Computer Maufacturer Association* (ECMA) mendatarkan beberapa tujuan desain dari bahasa pemrograman C#, sebagai berikut [10]:

1) Bahasa pemrograman C# dibuat sebagai bahasa pemrograman yang bersifat bahasa pemrograman general – purpose (untuk tujuan jamak), berorientasi objek, modern, dan sederhana.
2) Bahasa pemrograman C# ditujukan untuk digunakan dalam mengembangkan komponen perangkat lunak yang mampu mengambil keuntungan dari lingkungan terdistribusi.
3) Portabilitas programmer sangatlah penting, khususnya bagi programmer yang telah lama menggunakan bahasa pemrogaman C dan C++.
4) Dukungan untuk internasionalisasi (*multi- language*) juga sangat penting.

C# ditujukan agar cocok digunakan untuk menulis program aplikasi baik dalam sistem *klien-server* (*hosted system*) maupun sistem 38 *embedded* (*embedded system*), mulai dari perangkat lunak yang sangat besar yang menggunakan sistem operasi yang canggih hingga kepada perangkat lunak yang sangat kecil yang memiliki fungsi-fungsi terdedikasi.

2.3.9 SQL Server

SQL Server 2008 sangat terkenal sebagai Relational *Database Management System* atau RDBMS. Namun tidak hanya itu, tetapi lebih akurat digambarkan sebagai data *enterprise platform* yang dibangun di atas banyak fitur yang pertama kali didirikan di dalam SQL Server 2008. Sementara, SQL Server 2008 memperluas penawaran yang berguna untuk menyertakan beberapa perbaikan dan juga penambahan. Karena perannya sebagai RDBMS tradisional, SQL Server 2008 juga menyediakan kemampuan seperti *reporting, analisis data* dan *data mining*. SQL Server 2008 juga memiliki banyak fitur yang mendukung

aplikasi data yang berbeda, *data driven event notification* dan masih banyak lagi [11].

SQL Server 2008 diciptakan dengan berbagai edisi yang mempunyai tujuan yang berbeda-beda dan masing-masing edisi dari SQL Server 2008 juga memiliki tempat khusus dalam infrastruktur pengelolaan data, kecuali pada edisi *Enterprice Evaluation* yang hanya berfungsi untuk evaluasi produk jangka pendek selama 180 hari. Edisi-edisi yang berbeda pada tingkat teratas seperti edisi *Enterprise Edition*, di mana edisi tersebut benar-benar mendukung semua yang SQL Server 2008 tawarkan. Sebaliknya adalah *Express Edition*, di mana edisi tersebut menawarkan fitur-fitur yang terbatas [11]. Edisi-edisi yang tersedia seperti :

1. Enterprise edition
2. Standar edition
3. Developer edition
4. Compact edition
5. Express edition
6. Workgroup edition
7. Web edition