BAB II

TINJAUAN PUSTAKA

2.1 Profil Perusahaan

UD.Singapur Group adalah satu dari beberapa perusahaan yang bergerak dibidang jual beli dan distribusi sepatu . UD.Singapur Group didirikan pada tahun 1984 oleh Bapak H.Basri dan HJ.Nurhayati dengan nama Bintang Perca. Saat itu UD.Singapur Group merupakan toko Grosir pertama yang berada di wilayah Bukittinggi. Tahun 1989 “Bintang Perca” berganti nama menjadi “Singapur” yang berpusat di Aur Kuning. Pada saat itu pemilik perusahaan ingin membuat produk sendiri dan mendistribusikan produk tersebut ke toko-toko yang lain sehingga pada tahun 1996 Toko Singapur memperluas cabangnya yang berada di Pekanbaru-Riau dengan nama ‘Indah Singapur” guna memudahkan dalam pendistribusian barang kepada toko-toko yang berada di luar provinsi Sumatera Barat .

Setelah terjadi krisis Moneter UD.Singapur Group mengalami kendala dalam memproduksi produk sepatu, sehingga pemilik saat itu memutuskan untuk memasok barang dari pabrik-pabrik terdapat di luar Sumatera contohnya PD. Mitra Lestari dsb. Pada tahun 1997 Singpur mendirikan cabang yang lain yang bernama “Cahaya Singapur” guna mendistribusikan khusus produk-produk sepatu yang berasal dari pabrik.Semua cabang yang didirikan bergerak di bidang yang sama yaitu grosir sepatu, sehingga pada tahun 2000 Singapur, Cahaya Singapur dan Indah Singapur berada dibawah nama yang sama yaitu UD.Singapur Group

juga memperoleh ijin unit dagang No. 203/44.1/538.5.12/2000. Toko Utama UD.Singapur Group berada di Pusat Grosir Terminal Aur Kuning No.3 Bukittinggi-Sumatera Barat, dengan no.telepon : +0752-34970.

2.2 Landasan Teori

Subbab ini berisi teori-teori pendukung yang digunakan dalam proses analisis dan implementasi tugas akhir ini.

2.2.1 *Data warehouse*

Data warehouse adalah sebuah database yang mewakili sejarah bisnis suatu perusahaan atau organisasi. *Data warehouse* merupakan kombinasi dari *database* yang berbeda-beda. *Data warehouse* mengkombinasikan informasi dengan meringkas (*summarizing*) dan mengelompokkan (*aggregation).* Informasi yang diperlukan ini didefinisikan oleh persetujuan para pengguna sesuai dengan informasi yang mereka butuhkan dalam pengambilan keputusan. Jadi suatu *data warehouse* hanya berisi informasi yang relevan dengan kebutuhan *user* untuk mendukung pengambilan keputusan.

2.2.1.1 Pengertian *Data warehouse*

Pengertian Data Warehouse menurut beberapa ahli adalah sebagai berikut :

Menurut Reinardi dalam bukunya Building Data Warehouse in SQL Server 2008 (2008:1-3), data warehouse merupakan sistem yang mengambil (retrieve) dan mengkosilidasikan (Consolidate) data secara periodik (Periodically) dari sumber data ke dalam penyimpanan dimensional dan ternormalisasi. Data yang disimpan

merupakan data histori dan digunakan untuk permintaan data (query) pada bisnis intellegence dan aktifitas analisis.

Menurut W.H Inmon dalam bukunya Building The Data Warehouse (2005), data warehouse adalah koleksi data yang mempunyai sifat berorientasi subjek (subject-oriented) terintegrasi dan konsisten (Integrated and consistent), time-variant dan bersifat tetap dan tidak berubah (Not volatile).

Dari definisi-definisi diatas, dapat disimpulkan *data warehouse* adalah database yang saling bereaksi yang dapat digunakan untuk *query* dan analisisis, sedangkan menurut Bill Inmon *data warehouse* bersifat orientasi subjek, terintegrasi, time-variant,tidak berubah yang digunakan untuk membantu para pengambil keputusan. Adapun penjelasan diatas dapat dijelaskan sebagai berikut :

1. ***Subject-oriented***

   Data warehouse diorganisasikan dalam lingkup subjek Berfokus pada spesifikasi konsep bisnis, model dan analisis pada data untuk membuat keputusan. Contoh U.D Singapur Group, subyek yang terlibat antara lain pelanggan, barang, sales, transaksi, order,dan lainnya. Orientasi subjek memungkinkan para pengguna untuk menentukan tidak hanya bagaimana bisnis mereka sedang berjalan tetapi mengapa bisnis mereka berjalan. *Data warehouse* berbeda dengan database operasional dalam hal kebanyakan database operasional mempunyai sebuah orientasi produk dan disetel untuk menangani transaksi yang memperbaharui database. Orientasi subjek menyediakan pandangan yang menyeluruh mengenai organisasi dan

memisahkan data yang tidak berguna di dalam proses pengambilan keputusan.

2. ***Integrated***

   Data warehouse dibangun dengan menggabungkan/menyatukan data yang berbeda. Penggunaan penamaan, bentuk dan struktur yang konsisten serta karakteristik yang saling berhubungan dari berbagai sumber internal maupun eksternal perusahaan/organisasi.

3. ***Time-Variant***

   Data operasional biasanya memproteksi sebuah periode waktu yang singkat karena banyak transkasi berhubungan dengan beberapa data. Waktu adalah dimensi penting yang harus didukung oleh semua *data warehouse* misal harian, mingguan, bulanan, dimana setiap struktur kunci mengandung elemen waktu. Dimensi waktu merupakan dimensi yang dikandung data dalam *data warehouse* . Data disimpan untuk menyediakan informasi dari perspektif historical artinya data tidak menyediakan status saat ini. Mereka disimpan untuk lima atau sepuluh tahun atau lebih dan digunakan untuk mempelajari trend, peramalan dan perubahan dari informasi.

4. ***Nonvolatile***

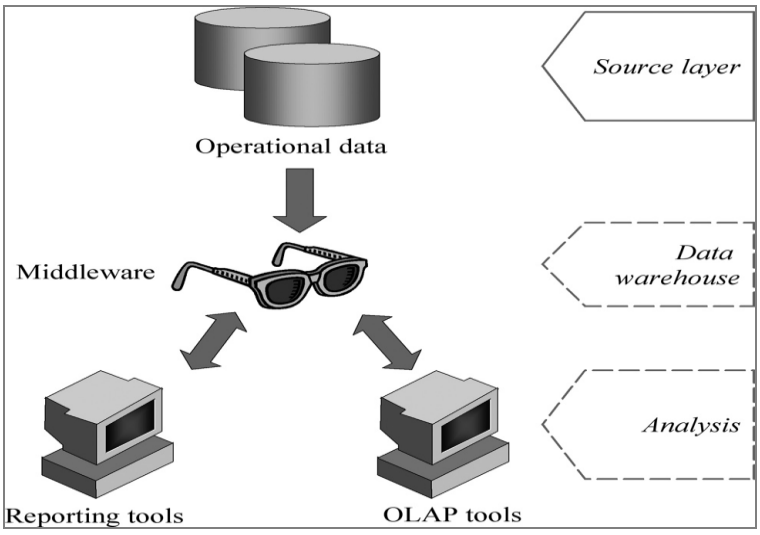
   Sekali dimasukkan kedalam *data warehouse* , data adalah *read-only*, mereka tidak bisa diubah atau diperbaharui. Data using dibuang dan perubahan direkam sebagai data baru. Hal ini memungkinkan *data warehouse* untuk disesuaikan hamper secara eksklusif untuk akses data.

2.2.1.2 Arsitektur *Data warehouse*

Menurut Golfarelli dan Rizzi dalam bukunya Data Warehouse Design, Modern Principles and Methodologies (2009:7-29), Terdapat 3 jenis arsitektur data warehouse yaitu Single-Layer Architecture, Two-Layer Architecture, dan Three-Layer Architecture. Adapun dari penjelasannya akan dijabarkan sebagai berikut :

1. ***Single Layer Architecture***

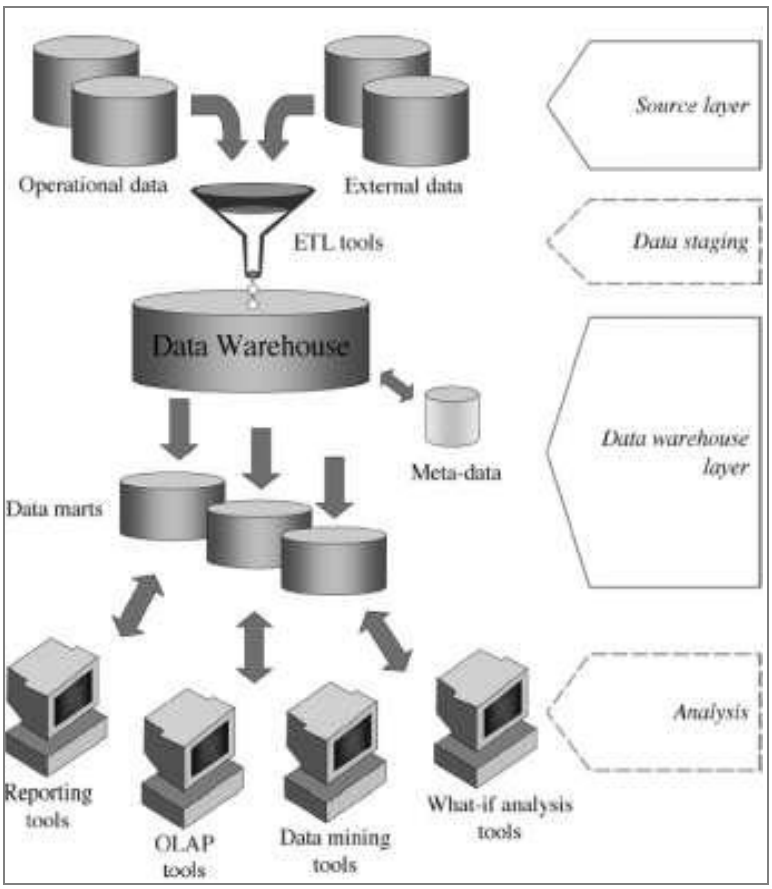
   Dalam *Single Layer Architecture* hasil akhir yang diinginkan adalah untuk memperkecil total keseluruhan data store, untuk mencapai hasil akhir tersebut data yang bersifat redudansi akan dihapus. Dalam kasus ini *data warehouse* bersifat virtual artinya *data warehouse* di implementasikan sebagai *dimensional view* dari data operasional dan dibuat sebagai *middleware.* Kelemahan dari arsitektur ini adalah kesalahan dalam menggabungkan antara analisis dan proses transaksi. Gambar 2.1 menjelaskan *Single-Layer Architecture* dalam *data warehouse* .



Gambar 2.1 *Single-Layer architecture*

2. Two-Layer Architecture

Dalam suatu perusahaan, data operasional biasanya berada pada daerah kekuasaan departemen masing-masing dalam bentuk *database OLTP*.. Gambar 2.2 Menjelaskan mengenai Two-Layer Architecture dalam *data warehouse* , dimana dalam arsitektur ini media penyimpanan antara sumber data dan *data warehouse* dipisahkan.



Gambar 2.2 *Two-Layer Architecture*

Lapisan pertama adalah *source layer*. Pada lapisan ini, data masih berupa operasional data, artinya *data warehouse* dibangun dengan mengintegrasikan data-data yang berasal dari berbagai sumber data, yaitu *database* operasional atau ekernal data lainnya.

Lapisan kedua adalah *data staging area*. Pada lapisan ini, data operasional akan diekstrak (lebih dikenal dengan proses ETL) ke dalam *data warehouse* . Sesuai dengan namanya, aplikasi ETL melakukan fungsi-fungsi *Extract, Transform,* dan *Load*. Proses *extract* adalah proses pengambilan data dari sumber data. Disebut *extract*, karena proses pengambilan data ini tidak mengambil data matang saja. Proses *extract* ini harus mengakomodir berbagai macam teknologi yang digunakan oleh sumber data dan diintegrasikan ke dalam database tunggal. Kemudian data hasil *extract* ini menjalani proses *transform* yang pada prinsipnya adalah mengubah kode-kode yang ada menjadi kode-kode standar, misalnya kode propinsi. Hal ini perlu dilakukan mengingat data-data yang diambil berasal dari sumber yang berbeda yang kemungkinan memiliki standardisasi yang berbeda pula. Standardisasi diperlukan untuk nantinya memudahkan pembuatan laporan. Proses *load* dalam *ETL* adalah suatu proses mengirimkan data yang telah menjalani proses transformasi ke gudang data akhir, yaitu data *warehouse* itu sendiri dimana aplikasi reporting dan *business intelligence* siap untuk diakses.

Lapisan ketiga adalah data *warehouse layer*. Informasi akan disimpan pada sebuah penyimpanan *logic* yang tersentralisasi, yaitu data *warehouse* . Data *warehouse* dapat diakses secara langsung, dan juga bisa digunakan sebagai sumber untuk membuat *data marts* yang merupakan sebagian dari duplikasi *data warehouse* dan dirancang khusus bagian khusus.

Lapisan keempat adalah *analysis*. Ada 4 tugas yang nantinya dapat dilakukan, keempat tugas tersebut yaitu [3]:

a. Pembuatan Laporan (*Reporting*)

Pembuatan laporan adalah salah satu tugas *data warehouse* yang paling umum dilakukan, dengan menggunakan query sederhana didapatkan laporan per-hari/minggu/bulan/tahun atau jangka waktu kapanpun yang diinginkan.

b. On-Line Analytical Processing (OLAP)

Dengan adanya *data warehouse* , semua informasi baik detail ataupun summary yang dibutuhkan dalam proses analisa mudah didapat. *OLAP* mendayagunakan konsep data multidemensi dan memungkinkan para pengguna menganalisa data sampai mendetail, tanpa mengetik satupun perintah SQL. Hal ini memungkinkan karena menggunakan konsep multidimensi (diintegrasikan kedalam tabel fakta dan tabel dimensi).

c. Data Mining

Data mining merupakan proses untuk menggali (mining) pengetahuan dari data yang berjumlah banyak pada *data warehouse* , dengan menggunakan kecerdasan buatan (Artificial Intelegence), static dan matematika. Data Mining merupakan teknologi yang diharapkan dapat menjebatani komunikasi antara data dan pelakunya.

d. Proses Informasi Executive

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

3. Three-Layer Architecture.

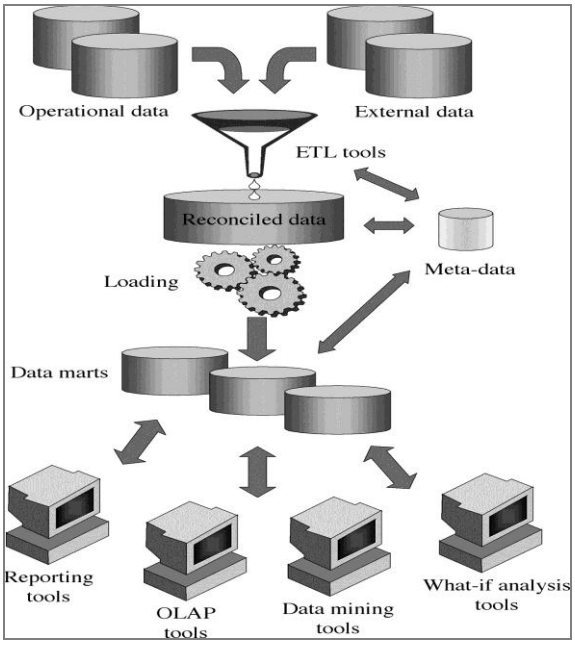
Perbedaan *Two-Layer Architecture* dengan *Three-Layer Architecture* yaitu terletak pada lapisan ketiga. Dalam arsitektur ini, lapisan ketiga adalah lapisan data *reconciled* atau operasional data store. Berikut penjelasan dari *Three Layer Architecture:*

1. Data Operasional disimpan dalam berbagai sistem operasional dalam setiap bagian organisasi. Lapisan ini terlaksana apabila data operasional yang diperoleh dari sumber data sudah terintegrasi dan bersih (telah melalui proses ETL) dan data-data yang ada dapat terintegrasi, konsisten, benar, mutakhir, dan rinci.

2. *Reconciled Data* atau *Operational Data Store (ODS)*

Keuntungan utama dari lapisan data *reconciled* adalah menciptakan model referensi umum data untuk keseluruhan perusahaan pada saat yang sama juga memisahkan masalah ekstraksi *data source* dan integrasi operational *data store.*

Penyimpanan data berada dalam gudang data perusahaan atau data mart. Tahap ini berfungsi untuk memasukkan data ke dalam *Enterprise Data warehouse* (EDW) yang terintegrasi secara terpusat dan tersedia untuk pendukung keputusan.. Gambar 2.3 Menjelaskan mengenai Three-Layer Architecture dalam *data warehouse* .

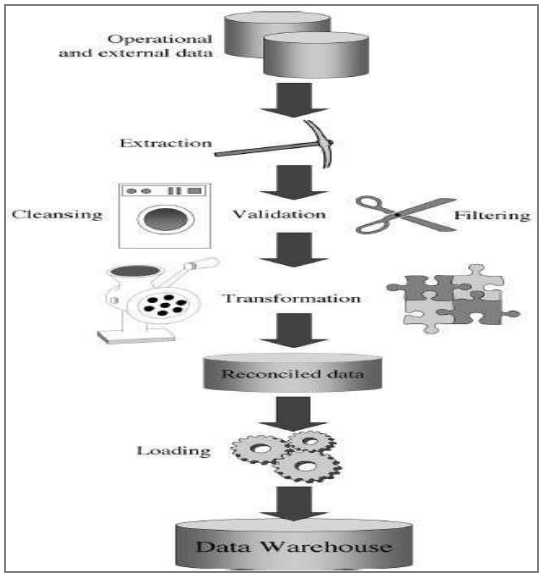


Gambar 2.3 *Three -Layer Architecture*

2.2.2 *Data Staging* dan *ETL*

Merupakan fase yang terjadi ketika mengintegrasikan data ke dalam *data warehouse* . Tiga fungsi utama yang perlu dilakukan untuk membuat data siap digunakan pada datawarehouse adalah *Extraction, Transformation* dan *Loading*. *Transformation* merupakan proses yang mempunyai peran dalam melakukan perubahan dan integrasi skema serta struktur yang berbeda-beda ke dalam skema dan struktur yang terdefinisi dalam *data warehouse* . Ketiga fungsi ini terdapat

pada staging area. Gambar 2.4 menjelaskan mengenai *data staging* dan proses ETL:



Gambar 2.4 *Data Staging* dan *ETL* dalam *Data warehouse*

1. ***Extraction***

   Bagian pertama dari suatu proses ETL adalah men-ekstrak data dari sumber data. Disebut ekstrak, karena proses mengubah data ke dalam suatu format yang berguna untuk proses transformasi. Pengambilan data ini tidak mengambil keseluruhan data yang ada di *database* operasional, melainkan hanya mengambil data matang saja. Pada proses ekstraksi data terdapat proses cleasning data dimana data operasional yang telah diambil/dibaca akan diperbaiki dari kesalahan- kesalahan pada input data, redudansi data, nilai field yang tidak konsisten, dan sebagainya. Proses ini meliputi penyaringan data yang digunakan dalam pembuatan *data warehouse* , dapat langsung dimasukkan langsung dalam *data warehouse* atau dimasukkan dalam tempat penampungan sementara

terlebih dahulu. Pada hakikatnya bagian dari ekstraksi melibatkan penguraian dari data yang telah diekstrak, menghasilkan suatu pengecekan jika data bertemu dengan suatu struktur atau pola yang diharapkan. Jika bukan, data tersebut mungkin ditolak secara keseluruhan.

2. ***Transformation***

Proses yang ke dua adalah transformasi data yang telah diekstrak ke dalam format yang diperlukan. Hal ini perlu dilakukan mengingat data yang diambil berasal dari sumber yang berbeda yang kemungkinan memiliki standardisasi yang berbeda pula. Data dari beberapa sistem perlu ditransformasi ke dalam format umum yang disepakati dan digunakan dalam *data warehouse* . Aturan- aturan data Transformation antara lain :

a. *Extracting*

Mengambil data dari sumber operasional dalam “as is” status, dan sumber-sumber data berasal dari mainframes yang hampir seluruhnya berupa database relasional

b. *Conditioning*

Perubahan tipe data dari sumber data ke target data (*data warehouse*) tetapi tetap dalam bentuk database relasional missal : dilakukan perubahan tipe data atau penamaan data field yang ada di database relasional ke dalam *data warehouse* dengan format yang dimengerti dan disesuaikan dengan kebutuhan.

c. *Scrubbing*

Digunakan untuk membersihkan/meningkatkan kualitas data.

d. *Merging*

Proses penggabungan data dengan memperhatikan aturan-aturan dalam mengintegrasikan data

e. *House Holding*

Mengidentifikasikan semua anggota dari household (berada di alamat yang sama) untuk mengirimkan report. Menjamin hanya satu jenis dokumen saja yang dikirimkan kepada setiap household karena mengirimkan multiple report akan mengakibatkan naiknya "*cost*" pada *data warehouse* .

f. *Enrichment*

Mengambil data dari sumber data eksternal (diluar data source yang seharusnya) untuk memperkaya data operasional yang sudah ada.

g. *Scoring*

Komputasi probabilitas tentang sebuah event, sebagai contoh menghitung kemungkinan pelanggan membeli produk baru.

h. *Delta Updating*

Proses update yang hanya mengupdate data baru saja tanpa menghasilkan duplikasi record.

i. Refresh

Proses yang mempropagasi terjadinya proses pengupdatean dari sumber data ke *data warehouse* , caranya misalnya melakukan refresh secara periodic (setiap malam/minggu) atau setelah event-event yang signifikan, ketika *data warehouse* membutuhkan data yang bersangkutan, atau berdasarkan kebijakan refresh yang dibuat oleh administrator berdasarkan kebutuhan pengguna.

j. Validating

Proses pemeriksaan kebenaran data yang diihasilkan dari penggabungan.

3. ***Load***

Data load adalah memindahkan data ke data*warehouse*. Ada dua loading data yang dilakukan pada data*warehouse*. Ada 2 cara dalam proses loading yaitu :

a. Refresh

Dalam proses refresh *data warehouse* secara keseluruhan ditulis ulang, artinya data yang lama diganti. Refresh biasanya digunakan dalam kombinasi ekstraksi statis untuk berada dalam area *data warehouse*

b. Update

Hanya data-data tertentu yang dirubah dan ditambahkan ke dalam *data warehouse* . Update biasa dilakukan tanpa menghapus atau memodifikasi data yang sudah sebelumnya. Teknik ini menggunakan

kombinasi dengan ekstraksi tambahan untuk memperbaharui *data warehouse* secara berkala.

2.2.3 Perancangan *Data warehouse*

2.2.3.1 Pendekatan Perancangan

Sebelum merancang *data warehouse* sebelumnya kita harus mengetahui pendekatan dalam perancangan dalam *data warehouse* yaitu :

1. **Pendekatan Normalisasi**

   Desain ternomalisasi menyediakan performansi yang optimal untuk system OLTP, karena banyaknya transaksi yang diperlukan untuk mengupdate data. Normalisasi memastikan tabel-tabel terbentuk secara benar dan menghindari terjadinya redudansi, dengan hanya memiliki satu copy data. Data dalam *data warehouse* disimpan berdasarkan aturan normalisasi data, Kelebihan dalam pendekatan normalisasi yaitu mudah untuk menambahkan data/informasi ke dalam database, tetapi memiliki kelemahan dimana banyak tabel yang terlibat sehingga akan menimbulkan kesulitan dalam penggabungan data.

2. **Pendekatan Dimensional**

   Untuk mengoptimalkan performasi warehouse, dimana aktifitas utama yang dilakukan adalah melakukan query atas data, maka diperlukan model data yang baru yang mendukung keperluan ini. Pada *data warehouse* digunakan teknik pemodelan data yang disebut *dimensional*

modelling technique. Pemodelan dimensional adalah suatu model berbasis pemanggilan yang mendukung akses *query* volume tinggi. Pendekatan dimensional pada pemodelan yaitu mengorganisasi data ke dalam tabel fakta (data transaksi numeric) dan tabel dimensi (referensi informasi dari fakta). Pendekatan dimensional lebih sering digunakan walaupun terkadang di dalam pendekatannya tetap membutuhkan normalisasi. Adapun kelebihan dari pendekatan dimensional adalah data direpresentasikan dalam bentuk yang lebih mudah dipahami oleh user. Sedangkan kekurangannya adalah susah untuk mempertahankan integritas dari fakta dan dimensi.

2.2.3.2 Multi-Dimensional Modelling

Data warehouse dan *OLAP* tools didasarkan pada model data multidimensional. Model ini memeandang data dalam bentuk kubus data, yaitu memungkinkan data yang dimodelkan dan dilihat dalam berbagai dimensi (N-dimensi) **.** Pemodelan data dimensional berguna untuk meringkas dan menyusun data dan memperlihatkan data untuk mendukung para analisis data[5].

Pendekatan model dimensional yaitu mengorganisasi data ke dalam bentuk fakta (*Fact*) diwakili dalam bentuk tabel fakta (diorganisir berdasarkan tema sentral) contohnya data transaksi penjualan. Hal-hal seperti data transaksi numeric seperti harga, jumlah, berat, dll disebut sebagai *measure*. Pendekatan lainnya yaitu mengorganisasi data ke dalam *dimension* (referensi informasi dari fakta) dalam bentuk tabel dimensi contoh: yang berkaitan dengan tabel transaksi

penjualan : produk, customer, dll. Penjelasan mengenai fakta, dimensi dan measure adalah sebagai berikut :

1. ***Fact***

   Fact adalah suatu koleksi dari relasi data-data *items*, terdiri dari ukuran-ukuran dan konteks data. Setiap *fact* biasanya merepresentasikan sebuah bisnis *item*, suatu transaksi bisnis, atau sebuah kejadian yang dapat digunakan dalam analisis bisnis atau proses bisnis. Dalam *data mart*, *fact* di implementasikan dalam tabel dasar dimana semudah data *numeric* dan disimpan[3].

2. ***Dimensions***

   Dimensions adalah suatu koleksi dari anggota atau unit-unit data dengan tipe yang sama. Dalam sebuah diagram, suatu dimensi biasanya direpresentasikan dengan suatu *axis.* Dalam dimensional model, semua data menunjukan *fact tabel* yang diasosiasikan dengan satu dan hanya satu member sari setiap *multiple dimensions.* Jadi dimensi menunjukan latar belakang kontekstual dari *fact.* Banyak proses analisis yang digunakan untuk menghitung (*quatify*) dampak dari dimensi pada *fact[3]* .

   Dimensi adalah parameter dari apa yang ingin lakukan dalam *On-Line Analytical Processing (OLAP)[2]* sebagai contoh, dalam suatu *database* untuk menganalisa semua penjualan dari produk, dimensi berikut ini harus ada : Waktu, Lokasi, Pembeli, dan Penjual

Dimensi biasanya juga dapat dipetakan bukan angka numerik, *entity* yang bersifat informatif seperti merek atau karyawan, berikut merupakan penjelasan :

a. ***Dimension member*** : suatu dimensi berisi banyak anggota-anggota. Suatu anggota dimensi adalah nama pembeda atau *indetifier* yang digunakan untuk membedakan posisi suatu data item. Sebagai contoh, semua bulan, empat bulanan dan tahunan membuat dimensi waktu. Dan semua kota, wilayah dan negara merupakan dimensi geografi[2].

b. ***Dimension Hierarchies*** *:* dapat mengatur anggota dari suatu dimensi kedalam satu atau banyak hirarki. Setiap hirarki dapat juga mempunyai banyak level hirarki. Setiap member dari dimensi tidak boleh dilokasikan kedalam satu struktur hirarki[2] .

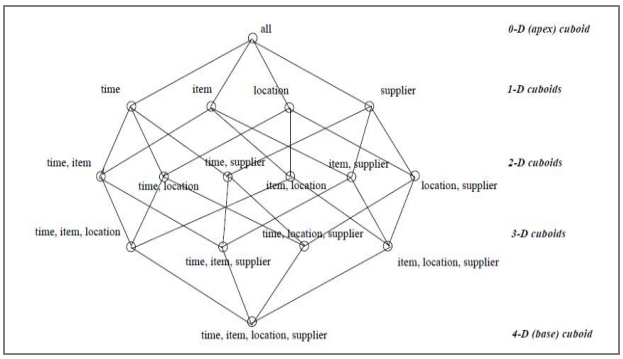
3. ***Measure***

Suatu *measures* (ukuran) adalah suatu besaran (angka numerik) atribut dari sebuah *fact*, yang menunjukan *performance* atau *behavior* (tingkah laku) dari bisnis secara relatif pada suatu dimensi [2]. Angka atau nomor yang ditunjukan disebut dengan *variable*. Sebagai contoh ukuran dari penjualan dalam bentuk uang, besarnya penjualan, jumlah pengadaan, biaya pengadaan, banyaknya transaksi dan lainnya. Suatu ukuran dijelaskan dengan kombinasi dari member dari suatu dimensi dan diletakkan dalam *fact.*

2.2.3.3 Visualisasi Dari Suatu Dimensional Model

Cara yang paling populer dalam memvisualisasikan suatu model dimensional adalah dengan menggambarkan sebuah *cube* data. Contohya dapat dilihat pada (Gambar 2.5) dapat menggambarkan tiga model dimensional menggunakan sebuah kubus. Biasanya suatu dimensional model terdiri dari lebih tiga dimensi dan digambarkan sebagai suatu *hypercube*, akan tetapi *hypercube* sulit untuk divisualisasikan, jadi sebuah kubus lebih biasa digunakan [2]. Gambar 2.5 menjelaskan bentuk data cube berupa:

a. Cube 1 dimensi yaitu : masing-masing dari dimensi waktu, item, location dan supplier
b. Cube 2 dimensi yaitu : time-item, time-supplier, item-supplier, location-supplier, time-location, item-location.
c. Cube 3 dimensi yaitu : time-item-location, time- location-supplier, time-item-supplier dan item-location-supplier.
d. Cube 4 dimensi dimana semua dimensi digabung menjadi 1 yaitu time, item, location, dan supplier.

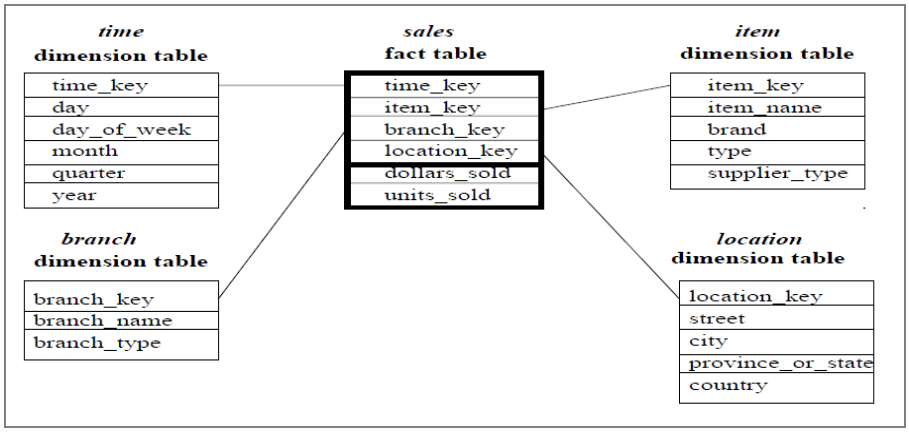


Gambar 2.5 Visualisasi dari model Dimensional

Terdapat tiga konsep yang dapat memodelkan lebih jelas dalam penentuan *fact tabel* dimensi dan *measure*, yaitu :

1. Star Schema

   Star Schema dalah model dasar dari pemodelan multidimensi yang dapat di lihat pada gambar 2.7. Model star memiliki satu tabel induk yang dinamakan tabel fakta dan kumpulan dari tabel-tabel kecil yang disebut tabel dimensi dimana tabel dimensi diwakili hanya satu tabel saja dan setiap tabel berisi set atribut serta disusun dalam pola-pola melingkar mengelilingi tabel fakta [4]. Contoh model star yang terdapat pada gambar 2.6 terdapat satu tabel fakta yaitu tabel sales yang dikelilingi oleh 4 tabel dimensi yaitu dimensi waktu, dimensi item, dimensi lokasi dan dimensi cabang dan terdapat measure dalam tabel sales yaitu dollar_sold dan units_sold.

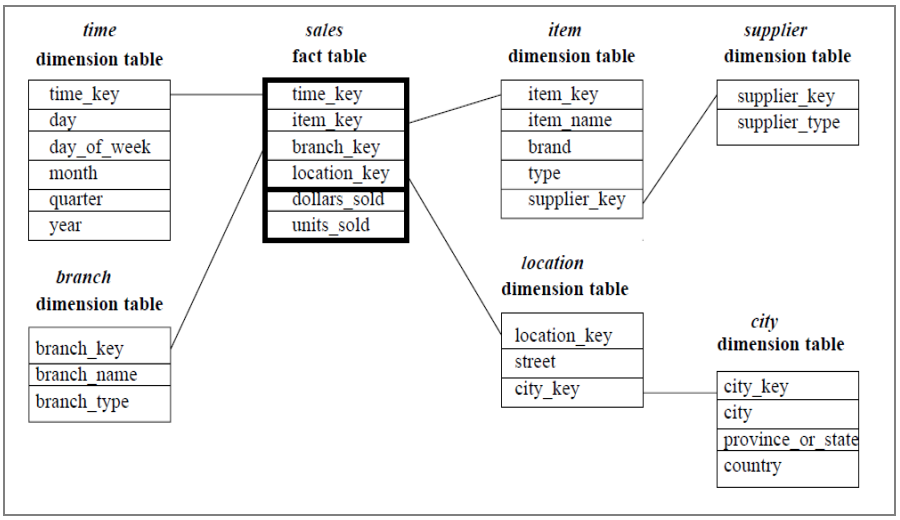


Gambar 2.6 Model *Star*

2. Model *SnowFlake*

   Model *SnowFlake* adalah variasi yang berbeda dari model star. Dalam sebuah model *snowFlake* terdapat sebuah tabel fakta yang dikelilingi oleh

beberapa tabel dimensi, namun tabel dimensi itu sendiri dapat seolah-olah berupa sebuah tabel fakta lain yang juga memiliki dimensinya sendiri [4] contohnya terdapat pada gambar 2.7 dimana pada model *snowflake* dapat dilihat bahwa tiap-tiap tabel dimensi berhubungan dengan dimensi yang lain seolah-olah tabel dimensi tersebut merupakan tabel fakta. Hal ini dapat terjadi karena pada model *snowflake* ini telah dilakukan normalisasi. Normalisasi ini tidak terdapat pada model star. Berdasarkan gambar 2.8 tabel sales merupakan tabel fakta yang dikelilingi oleh dimensi waktu, cabang, item, dan lokasi. Tetapi dimensi item dan dimensi lokasi di normalisasi sehingga memiliki turunan yaitu dimensi supplier dan dimensi kota.

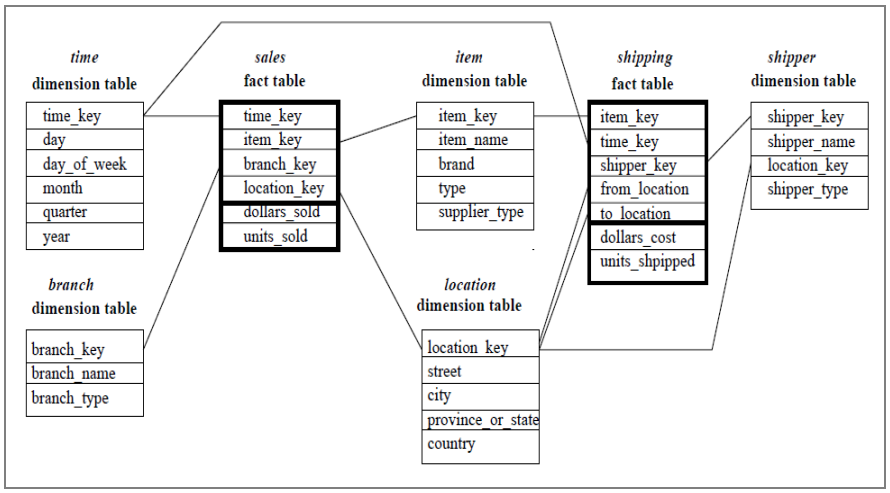


Gambar 2.7 Model *SnowFlake*

3. Model *Constellation.*

Model constellation ditampilkan sebagai koleksi dari kumpulan skema bintang atau sering disebut skema *galaxy[9]* . Aplikasi canggih mungkin memerlukan beberapa tabel fakta untuk berbagi tabel dimensi, contohnya

terdapat dalam gambar 2.8 Dalam data *warehouse* subjek-subjek dikumpulkan berupa informasi yang mencakup semua informasi dalam perusahaan seperti penjualan, pelanggan, barang, dll. Dalam *constellation* skema tabel fakta berbagi tabel dimensi dan tabel dimensi tersebut di normalisasi lagi sehingga tabel dimensi tersebut memiliki turunan tabel dimensi lainya. Pada gambar 2.8 terdapat 2 buah tabel fakta yaitu tabel sales dan tabel shipping dimana tabel tersebut dikelilingi oleh tabel dimensi yaitu dimensi waktu, cabang, item, pengirim dan lokasi. Dimensi lokasi, waktu dan item berbagi dengan 2 tabel fakta yang ada, sedangkan dimensi pengirim memiliki turunan tabel dimensi yaitu dimensi lokasi [9]



Gambar 2.8 Model *Constellation*

2.2.4 *OLAP*

OLAP digunakan untuk memproses informasi dan menampilkanya dalam bentuk multidimensi. *OLAP* memberikan jawaban terhadap query analytic untuk data multidimensional. Berikut penjelasan mengenai kubus *OLAP* dan teknik dalam penyimpanan *OLAP*.

2.2.4.1 *OLAP Cube* (Kubus *OLAP*)

Objek utama yang disimpan dalam sebuah basis data *OLAP* ialah *cube* (kubus). Sebuah kubus merupakan representasi multidimensi dari sekumpulan data, yang mengandung data secara *detail* maupun rangkumannya. Sebuah basis data *OLAP* dapat memiliki beberapa buah kubus sesuai dengan yang dibutuhkan, yang menggambarkan data yang ada dalam *data warehouse* . Sebuah kubus dibangun menggunakan dua komponen utama yaitu ukuran (*measure*) dan dimensi. Ukuran merupakan nilai numerik dari *fact tabel* dalam *data warehouse* seperti harga dari unit maupun kualitas dari *item*. Sedangkan dimensi menggambarkan kategori dari ukuran yang ada, seperti bagaimana ukuran berubah setiap waktu.

Dengan adanya *data warehouse* , semua informasi baik detail ataupun summary yang dibutuhkan dalam proses analisa mudah didapat. *OLAP* mendayagunakan konsep data multidemensi dan memungkinkan para pengguna menganalisa data sampai mendetail, tanpa mengetik satupun perintah *SQL*. Hal ini memungkinkan karena menggunakan konsep multidimensi (diintegrasikan kedalam tabel fakta dan tabel dimensi). Operasi yang terdapat dalam *OLAP* yaitu :

a. Rool-up dan *Drill Drown*

Consolidation atau yang lebih dikenal *roll up* merupakan operasi yang melakukan agregasi pada kubus data. Pada operasi *Roll Up* data bisa dilihat secara global atau berupa rangkuman (*summary*). *Drilling down* merupakan teknik untuk memecahkan sebuah informasi menjadi beberapa informasi yang lebih *detail*. Sebagai contoh, jika dilakukan *drill down* terhadap data tahunan maka akan dapat diperoleh data dalam catur wulan atau tri wulan [4].

b. *Slice* dan *Dice*

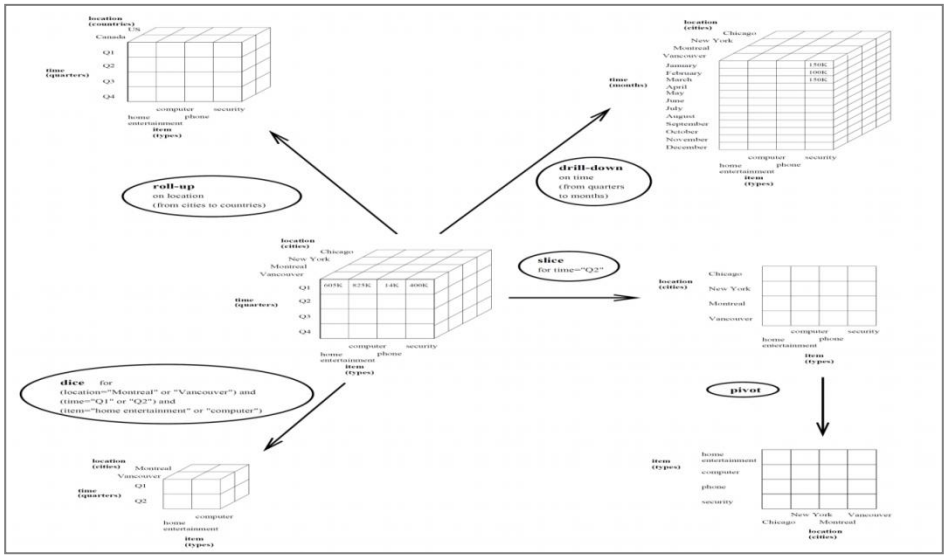
Slicing dan *dicing* adalah operasi untuk melihat data sebagai visualisasi dari kubus. Pengguna dapat mengekstrak bagian dari data *agregrated* dan dapat memeriksa dengan detail berdasarkan dimensi-dimensi yang diinginkan. Data *Agregrated* merupakan data praperhitungan (precalculated) dalam bentuk rangkuman data (data summarized) sehingga query pada kubus (cube) lebih cepat. *Slicing* memotong kubus sehingga dapat memfokuskan pada perspektif yang spesifik (pada suatu dimensi). Sedangkan *dicing* memberikan kemampuan untuk melihat pemilihan data pada dua dimensi atau lebih. Yaitu dengan merotasi cube pada perspektif yang lain sehingga pengguna dapat melihat lebih spesifik terhadap data yang dianalisa [4].

c. *Pivot*

Menampilkan nilai-nilai ukuran dalam tata letak tabel yang berbeda. Mengatur kembali dimensi dalam sebuah kubus data. *Pivotting* merupakan suatu teknik untuk saling menukarkan dimensi data. Dengan

melakukan *pivotting* maka dapat diamati suatu informasi atau data dari sudat pandang yang berbeda, sehingga diharapkan akan dapat memperjelas analisis yang dilakukan.

Gambar 2.9 menjelaskan masing-masing operasi yang terdapat dalam *OLAP*

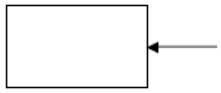
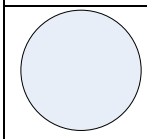


Gambar 2.9 Operasi-operasi yang terdapat dalam *OLAP*

2.2.5 *Data Flow Diagram*

Data Flow Diagram (DFD) adalah alat pembuatan model yang memungkinkan profesional sistem untuk menggambarkan sistem sebagai suatu jaringan proses fungsional yang dihubungkan satu sama lain dengan alur data, baik secara manual maupun komputerisasi. DFD ini adalah salah satu alat pembuatan model yang sering digunakan, khususnya bila fungsi-fungsi sistem merupakan bagian yang lebih penting dan kompleks dari pada data yang dimanipulasi oleh sistem. Dengan kata lain, DFD adalah alat pembuatan model yang memberikan penekanan hanya pada fungsi sistem.

DFD ini merupakan alat perancangan sistem yang berorientasi pada alur data dengan konsep dekomposisi dapat digunakan untuk penggambaran analisa maupun rancangan sistem yang mudah dikomunikasikan oleh profesional sistem kepada pemakai maupun pembuat program. Komponen DFD menurut Yourdan dan De Marco:

Nama	Keterangan
	Terminator Sumber (*source*) : terminator yang menjadi sumber.
	Terminator Tujuan (*sink*) : terminator yang menjadi tujuan data/informasi sistem.
	Proses : menggambarkan bagian dari sistem yang mentransformasikan *input* menjadi *output*.
	Data *store* : biasanya berkaitan dengan penyimpanan-penyimpanan, seperti *file* atau *database* yang berkaitan dengan penyimpanan secara komputerisasi, misalnya *file* disket, *file* harddisk, *file* pita magnetik. Data *store* juga berkaitan dengan penyimpanan secara manual seperti buku alamat, *file folder*, dan agenda.
	Alur data ini digunakan untuk menerangkan perpindahan data atau paket data/informasi dari satu bagian sistem ke bagian lainnya.

2.2.6 Flowchart

Flowchart merupakan representasi grafik dari algritma dengan menggunakan simbol-simbol tertentu yang masing-masing mempunyai fungsi

yang khusus [7], untuk lebih jelas dapat di lihat pada tabel 2.1. *Flowchart* menggambarkan atau langkah dari sistem (proses, operasi, fungsi atau aktifitas). *Flowchart* mempunyai beberapa simbol yang digunakan .

2.2.7 SQL Server 2008

SQL Server 2008 adalah sebuah terobosan baru dari Microsoft dalam bidang database. SQL Server adalah sebuah DBMS (Database Menagement System) yang dibuat oleh Microsoft untuk ikut berkecimpung dalam persaingan dunia pengolahan data menyusul pendahulunya seperti IBM dan Oracle. SQL Server 2008 dibuat pada saat kemajuan dalam bidang hardware sedemikian pesat.

SQL (*Structured Query Language*) adalah bahasa standar yang digunakan untuk mengakses *server database*. Semenjak tahun 70-an bahasa ini telah dikembangkan oleh IBM, yang kemudian diikuti dengan adanya *Oracle, Informix dan Sybase*. Dengan menggunakan SQL, proses akses *database* menjadi lebih *user-friendly* dibandingkan dengan misalnya *dBase* ataupun *Clipper* yang masih menggunakan perintah – perintah pemrograman murni.

Microsoft SQL Server adalah sebuah *server database* SQL *multiuser* dan *multi-threaded*. SQL sendiri adalah salah satu bahasa *database* yang paling populer di dunia. Implementasi program *server database* ini adalah program daemon 'mysqld' dan beberapa program lain serta beberapa pustaka. Sebagaimana *database* sistem yang lain, dalam SQL juga dikenal hierarki server dengan database-database. Tiap-tiap *database* memiliki tabel-tabel. Tiap-tiap tabel memiliki *field-field*. Umumnya informasi tersimpan dalam tabel – tabel yang

secara logik merupakan struktur 2 dimensi terdiri atas baris dan kolom. *Field-field* tersebut dapat berupa data seperti *int, realm char, date, time* dan lainnya. SQL tidak memiliki fasilitas pemrograman yang lengkap, tidak ada *looping* ataupun percabangan. Sehingga untuk menutupi kelemahan ini perlu digabung dengan bahasa pemrograman semisal *Pascal*.

2.2.8 Konsep Pemrograman Borland Delphi

Borland Delphi merupakan suatu bahasa pemograman yang memberikan fasilitas pembuatan aplikasi visual [6]. Bahasa pemograman yang digunakan adalah bahasa pemograman pascal atau kemudian disebut sebagai bahasa pemograman Delphi. Delphi sendiri merupakan penerus dari turbo pascal. Borland Delphi merupakan suatu bahasa pemrograman yang memberikan berbagai fasilitas pembuatan aplikasi visual. Bahasa pemrograman yang digunakan adalah bahasa pemrograman Pascal atau kemudian juga yang disebut bahasa pemrograman Delphi. Delphi merupakan generasi penerus dari Turbo Pascal. Delphi adalah compiler (penterjemah) bahasa Delphi (awalnya dari pascal) yang merupakan bahasa tingkat tinggi sekelas dengan basic, C. Bahasa pemrograman di Delphi disebut bahasa procedural yaitu bahasa atau sintaknya mengikuti urutan tertentu.

Delphi disebut juga Visual Programming artinya komponen-komponen yang ada tidak hanya berupa teks tetapi muncul berupa gambar-gambar. Delphi memiliki sarana untuk pembuatan aplikasi, mulai dari sarana untuk pembuatan form, menu, toolbar, hingga kemampuan untuk menangani pengelolaan basis data yang besar. Kelebihan-kelebihan yang dimiliki Delphi antara lain karena pada Delphi, form dan komponen-komponennya dapat dipakai ulang dan

dikembangkan, tersedia template aplikasi dan template form, memiliki lingkungan pengembangan visual yang dapat diatur sesuai kebutuhan, menghasilkan file terkompilasi yang berjalan lebih cepat, serta kemampuan mengakses data dari bermacam-macam format. Kombinasi ini menghasilkan sebuah lingkungan pengembangan aplikasi yang berorientasi objek (Object Oriented Programming). Dengan konsep seperti ini, maka pembuatan aplikasi menggunakan Delphi dapat dilakukan dengan cepat dan menghasilkan aplikasi yang tangguh. Form dan komponen yang ada didalamnya, misalnya, dapat disimpan dalam suatu paket komponen yang dapat digunakan kembali, atau dimodifikasi seperlunya saja. Khususnya untuk pemrograman database, Delphi menyediakan object yang sangat kuat, canggih dan lengkap, sehingga memudahkan pemrograman dalam merancang, membuat dan menyelesaikan aplikasi database yang diinginkan. Selain itu, Delphi juga dapat menangani data dalam berbagai format database, misalnya format MS.Access, Oracle, Foxro, Informix dan lain-lain. Format database yang dianggap asli dari Delphi adalah Paradox dan dBase.