BAB 2
TINJAUAN PUSTAKA

2.1 Profil Prusahaan

CV.Widuri merupakan perusahaan yang bergerak di bidang industri makanan ringan yang didirikan pada tahun 2005 oleh Bapak Dudung tobi budin. CV.Widuri beralamat di Kec. Kawali Kab. Ciamis. Desa Ciakar Cipaku – Ciamis. Dalam menjalankan usahanya CV.Widuri memasarkan makanan ringan ke kota – kota besar. Sebagai penjual makanan ringan CV.Widuri terus meningkatkan perusahaannya dengan menambah jenis – jenis makanan ringannya dan terus menambah jumlah toko – toko yang di suplay oleh CV.Widuri.

2.1.1 Visi dan Misi

Visi

Menjadi perusahaan penghasil makanan ringan yang bermutu tinggi, dan mampu menguasai pasar.

Misi

Menjadi perusahaan makanan ringan terbesar di Indonesia

2.1.2 Logo Perusahaan

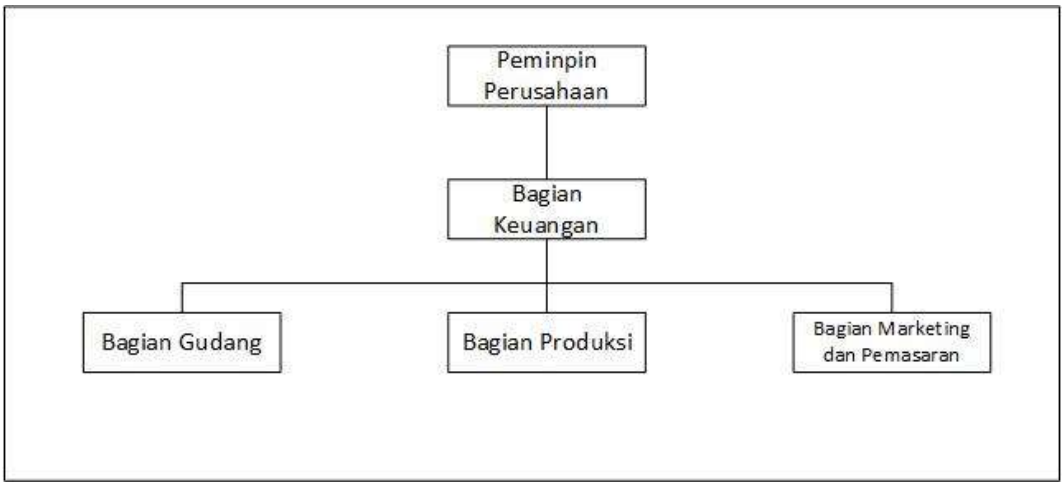
Adapun logo perusahaan di CV.Widuri adalah sebagai berikut :



Gambar 2.1 Logo CV.Widuri

2.1.3 Struktur Perusahaan

Struktur organisasi dalam suatu perusahaan merupakan hal yang sangat penting, dengan adanya struktur organisasi ini memberikan pembagian tugas sesuai dengan bidangnya masing-masing. Adapun struktur organisasi di CV.Widuri adalah sebagai berikut :



Gambar 2.2 Struktur Organisasi CV.Widuri

Tugas masing – masing pada struktur organisasi CV.Widuri

1. Tugas peminpin perusahaan adalah memantau dan menilai keseluruhan dari kemajuan perusahaan serta bertanggung jawab untuk pengambilan keputusan bagi perusahaan.
2. Tugas bagian keuangan adalah melaksanakan kegiatan administrasi keuangan perusahaan untuk menjamin kelancaran, keakuratan dan ketertiban administrasi keuangan perusahaan.
3. Tugas bagian gudang adalah sebagai penanggung jawab dari semua transaksi yang terjadi di gudang perusahaan.
4. Tugas bagian produksi adalah membuat produk dari mulai mengolah bahan mentah sampai berupa makanan jadi.

5. Tugas bagian marketing dan penjualan adalah mengatur jadwal penjualan dan melakukan penjualan untuk memenuhi permintaan pelanggan serta mencatat semua transaksi penjualan.

2.2 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 dari studi literatur yaitu pengumpulan data dengan cara mempelajari jurnal, browsing internet, referensi buku dan bacaan-bacaan lainnya yang berhubungan erat dengan penelitian yang dilakukan.

2.2.1 Data

Data adalah fakta mentah mengenai orang, tempat, kejadian, dan hal-hal yang penting dalam organisasi **[1]**.

Data merupakan bahasa, simbol-simbol pengganti lain yang disepakati oleh umum dalam menggambarkan objek, manusia, peristiwa, aktivitas, konsep. Singkatnya data merupakan suatu kenyataan apa adanya **[1]**.

2.2.2 Database

Database atau Basis data merupakan kumpulan data yang disimpan secara sistematis didalam komputer dan dapat diolah atau dimanipulasi menggunakan perangkat lunak (program aplikasi) untuk menghasilkan informasi. Pendefinisian basis data meliputi spesifikasi berupa tipe data, struktur, dan juga batasan-batasan data yang akan disimpan. Basis data merupakan aspek yang sangat penting dalam sistem informasi dimana basis data merupakan gudang penyimpanan data yang akan diolah lebih lanjut. Basis data menjadi penting karena dapat menghidari duplikasi data, hubungan antar data yang tidak jelas, organisasi data, dan juga update yang rumit **[1]**.

Basis data (Database) sendiri dapat didefinisikan dalam sejumlah sudut pandang seperti **[1]**:

1. Himpunan kelompok data (arsip) yang saling berhubungan yang diorganisasi sedemikian rupa agar kelak dapat dimanfaatkan kembali dengan cepat dan mudah.
2. Kumpulan data yang saling berhubungan yang disimpan secara bersama sedemikian rupa dan tanpa pengulangan (redun-dansasisi) yang tidak perlu, untuk memenuhi berbagai kebutuhan.
3. Kumpulan file, table, arsip yang saling berhubungan yang disimpan dalam media penyimpanan elektronis.

2.2.3 *Database Management System* (DBMS)

Aplikasi yang digunakan untuk membangun sebuah sistem basis data yang memungkinkan Pengguna untuk mendefinisikan, membuat, memelihara dan mengontrol akses ke database merupakan pengertian dari *Database Management System* (DBMS). DBMS membantu dalam pemeliharaan dan pengolahan kumpulan data dalam jumlah besar, sehingga dengan bantuan DBMS tidak akan menimbulkan kekacauan dan dapat digunakan oleh pengguna sesuai dengan kebutuhan. Bahasa basis data biasanya terdiri atas perintah-perintah yang diformulasikan dalam bentuk query sehingga perintah tersebut akan diproses oleh DBMS **[1]**.

2.2.4 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* **[2]**.

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 **[2]**.

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 **[2]**.

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.

2.2.4.1 Kegunaan *Data Warehouse*

Berdasarkan pengertian *data warehouse* diatas, *data warehouse* diperlukan bagi para pengambil keputusan manajemen dari suatu organisasi/perusahaan. Dengan adanya *data warehouse*, akan mempermudah pembuatan aplikasi-aplikasi DSS (*Decision Support System*) dan EIS (*Executive Information System*) karena kegunaan dari *data warehouse* adalah khusus untuk membuat suatu *database* yang dapat digunakan untuk mendukung proses analisa (OLAP), mengambil keputusan, pembuatan laporan, penggalian informasi baru (Data Mining) dari banyak data dan proses *executive* informasi **[3]**.

Beberapa kegunaan data warehouse:

a. Pembuatan laporan

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

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

   Dengan adanya *data warehouse*, semua informasi baik detail maupun hasil *summary* yang dibutuhkan dalam proses analisa mudah didapat. OLAP mendayagunakan konsep *data multidimensi* dan memungkinkan para pengguna menganalisa data sampai mendetail, tanpa mengetikkan satupun

perintah SQL. Hal ini dimungkinkan karena pada konsep *multidimensi*, maka data yang berupa fakta yang sama bisa dilihat dengan menggunakan fungsi yang berbeda. Fasilitas lain yang ada pada *sofware* OLAP adalah fasilitas *rool-up* dan *drill-down*. *Drill-down* adalah kemampuan untuk melihat detail dari suatu informasi dan *roll-up* adalah kebalikannya.

c. Data mining

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

Beberapa solusi yang diberikan data mining antara lain :

1. Menebak target pasar

   Data mining dapat mengelompokkan (*clustering*) model-model pembeli dan melakukan *klasifikasi* terhadap setiap pembeli dan melakukan *klasifikasi* terhadap setiap pemebeli sesuai dengan karakteristik yang diinginkan.

2. Melihat pola beli dari waktu ke waktu

   Data mining dapat digunakan untuk melihat pola beli dari waktu ke waktu.

3. *cross-market analysis*

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

4. Profil pelanggan

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

5. Informasi summary

   Data mining dapat membuat laporan *summary* yang bersifat *multidimensi* dan dilengkapi dengan informasi statistik lainnya.

6. Proses informasi *executive*

   Data warehouse dapat membuat ringkasan informasi yang penting dengan tujuan membuat keputusan bisnis, tanpa harus menjelajahi keseluruhan data.

Dengan menggunakan *data warehouse* segala laporan telah diringkas dan dapat pula mengetahui segala rinciannya secara lengkap, sehingga mempermudah proses pengambilan keputusan. Informasi dan data pada laporan *data warehouse* menjadi target *informative* bagi *user*.

2.2.4.2 Konsep Dasar *Data Warehouse*

Data warehouse adalah kumpulan macam-macam data yang *subject oriented, integrated, time variant*, dan *nonvolatile* dalam mendukung proses pembuatan keputusan **[4]**.

Data warehouse sering diintegrasikan dengan berbagai sistem aplikasi untuk mendukung proses laporan dan analisis data dengan menyediakan data histori, yang menyediakan infrastruktur bagi EIS dan DSS.

a. *Subject Oriented*

   Data warehouse diorganisasikan pada subjek-subjek utama, seperti pelanggan, barang, dan penjualan. Berfokus pada model dan analisis pada data untuk membuat keputusan, jadi bukan pada setiap proses transaksi atau bukan pada OLTP. Menghindari data yang tidak berguna dalam mengambil suatu keputusan.

b. *Integrated*

   Dibangun dengan menggabungkan atau menyatukan *data* yang berbeda. *relational databse*, *flat file*, dan *on-line transaction record*. Menjamin konsistensi dalam penamaan, struktur pengkodean, dan struktur atribut diantara data satu sama lain.

c. *Datawarehouse time variant*

   Data disimpan untuk menyediakan informasi dari *perspektif historical*, *data* yang tahun - tahun lalu atau 4 - 5 tahun. Waktu adalah elemen kunci dari suatu *data warehouse* pada saat pengcapture-an.

d. *Non Volatile*

   Setiap kali proses perubahan, data akan di tampung dalam tiap-tiap waktu. Jadi tidak di perbaharui terus menerus. *Data warehouse* tidak memerlukan

pemrosesan transaksi dan *recovery*. Hanya ada dua operasi *initial loading of data* dan *access of data*.

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

Tiga fungsi utama yang perlu dilakukan untuk membuat data siap digunakan pada *data warehouse* adalah *extraction, transformation* dan *loading*. Ketiga fungsi ini terdapat pada *staging area* **[5]**.

Pada *data staging* ini, disediakan tempat dan area dengan beberapa fungsi seperti data *cleansing, change, convert*, dan menyiapkan data untuk disimpan serta akan digunakan dalam *data warehouse* **[5]**.

a. *Extraction*

Data Extraction adalah proses pengambilan data yang diperlukan dari sumber *data warehouse* dan selanjutnya dimasukkan pada staging area untuk diproses pada tahap berikutnya. Pada fungsi ini banyak berhubungan dengan berbagai tipe sumber data seperti: *Format data*, mesin yang berbeda, *software* dan arsitektur yang tidak sama. Sehingga sebelum proses ini dilakukan, sebaiknya perlu didefinisikan requirement terhadap sumber *data* yang yang akan digunakan untuk proses berikutnya.

b. *Transformation*

Pada kenyataannya, pada proses transaksional data disimpan dalam berbagai format sehingga jarang kita temui data yang konsisten antara *aplikasi-aplikasi* yang ada. Transformasi data ditujukan untuk mengatasi masalah ini. Dengan proses transformasi data ini, kita melakukan standarisasi terhadap data pada satu format yang konsisten. Beberapa contoh ketidakkonsistenan *data* tersebut dapat diakibatkan oleh tipe data yang berbeda, *data length* dan lain sebagainya.

c. *Load*

Data load adalah memindahkan data ke *data warehouse*. Ada dua *loading data* yang dilakukan pada *data warehouse*. Pertama adalah *inisial load*, proses ini dilakukan pada saat telah selesai mendisain dan membangun *data warehouse*. Data yang dimasukkan akan sangat besar dan memakan waktu yang relatif lebih

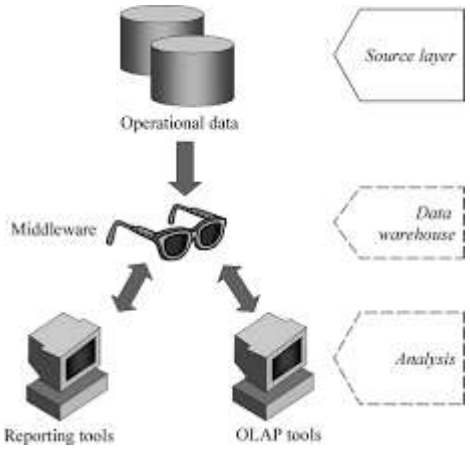
lama. Kedua *Incremental load*, dilakukan ketika *data warehouse* telah dioperasikan. *Incremental load* ini dapat dilakukan sesuai dengan sistem yang dibangun

2.2.4.4 Arsitektur *Data Warehouse*

Data Warehouse Design, Modern Principles and Methodologies, Terdapat 3 jenis arsitektur *data warehouse* yaitu *Single-Layer Architecture, Two-Layer Architecture*, dan *Three-Layer Architecture*. Adapun dari penjelasannya akan dijabarkan sebagai berikut : **[6]**

a. *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 2.3 menjelaskan *Single-Layer Architecture* dalam *data warehouse*.

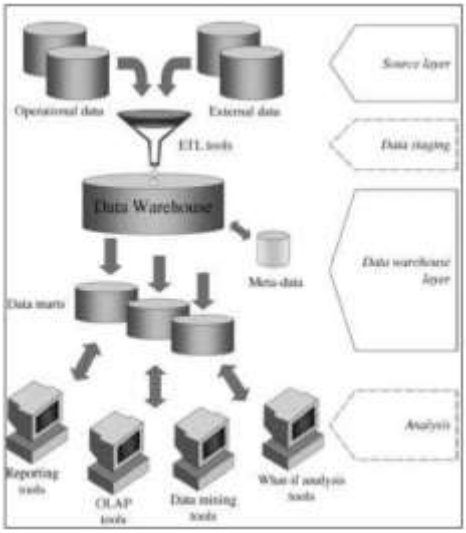


Gambar 2.3 Single-Layer architecture

b. *Two-Layer Architecture*

Dalam suatu perusahaan, data operasional biasanya berada pada daerah kekuasaan departemen masing-masing dalam bentuk *database* OLTP.. Gambar 2.4 Menjelaskan mengenai *Two-Layer Architecture* dalam *data warehouse,*

dimana dalam arsitektur ini media penyimpanan antara sumber *data* dan *data warehouse* dipisahkan.



Gambar 2.4 Two-Layer Architecture

a) 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.
b) 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.

c) 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.

d) Lapisan keempat adalah *analysis*. Ada 4 tugas yang nantinya dapat dilakukan, keempat tugas tersebut yaitu :

1. 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.

2. *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 m*ultidimensi* (diintegrasikan kedalam tabel fakta dan tabel dimensi).

3. *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.

4. *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.

c. *Three-Layer Architecture*.

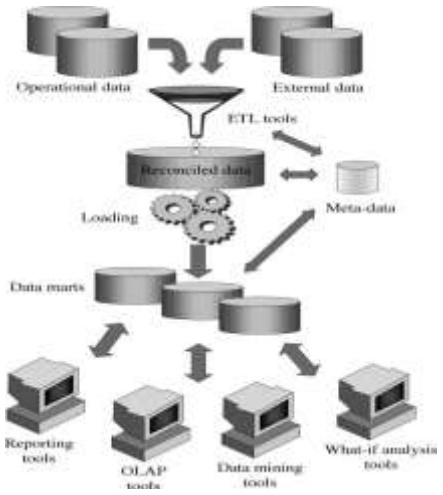
Perbedaan *Two-Layer Architecture* dengan *Three-Layer Architecture* yaitu terletak pada lapisan ketiga. Dalam arsitektur ini, lapisan ketiga adalah lapisan *data reconciled* atau *operasional data store*. Berikut penjelasan dari *Three Layer Architecture:*

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 2.5 Menjelaskan mengenai *Three-Layer Architecture* dalam *data warehouse.*



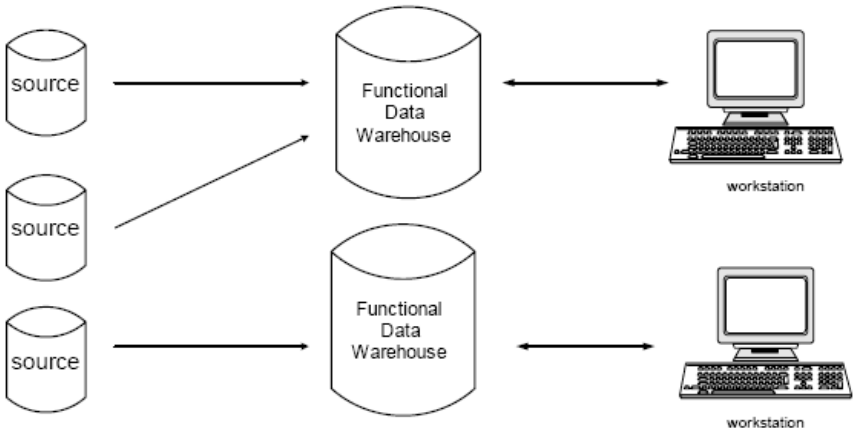
Gambar 2.5 Three-Layer Architecture

2.2.4.5 Jenis Dasar *Data Warehouse*

Jenis *data warehouse* yang digunakan oleh perusahaan akan berbeda-beda, *data warehouse* yang dibangun harus disesuaikan dengan sumber data dan kebutuhan perusahaan tersebut. Berikut jenis dasar *data warehouse*: **[6]**

1) *Functional Data Warehouse (Data Warehouse Fungsional*)

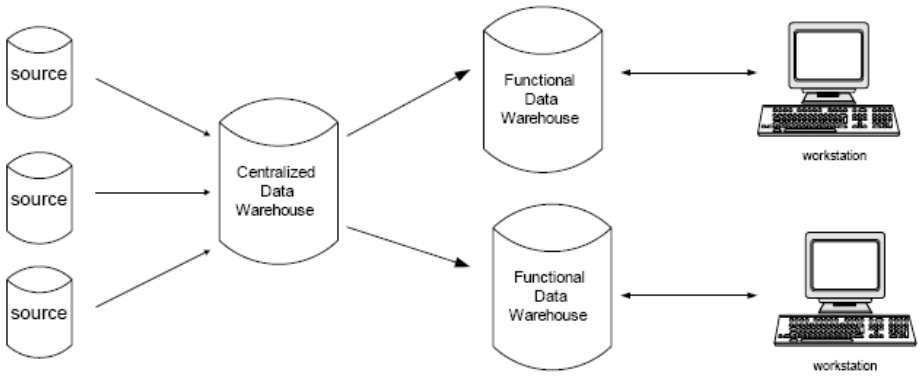
Data base yang digunakan jenis *data warehouse* ini adalah *database* yang diperoleh dari kegiatan sehari-hari. *Data warehouse* dibuat lebih dari satu dan dikelompokkan berdasar fungsi-fungsi yang ada di dalam perusahaan seperti fungsi keuangan (*financial*), marketing, personalia dan lain-lain. Keuntungan 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 2.6 Data Warehouse Fungsional

2) *Centralized Datawarehouse (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.

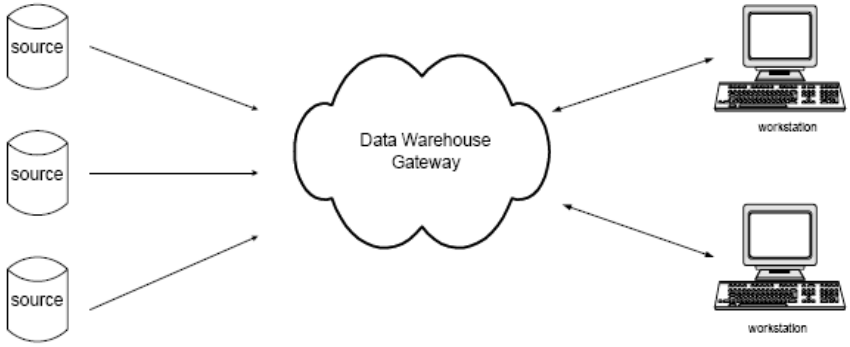


Gambar 2.7 Data Warehouse Terpusat

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 2.8 Data Warehouse Terdistribusi

2.2.4.6 Metodologi Perancangan Database untuk Data Warehouse

Sembilan tahap metodologi dalam perancangan *database* untuk *data warehouse*, yaitu : **[7]**

Langkah 1 : Pemilihan proses

a. *Data mart* yang pertama kali dibangun haruslah *data mart* yang dapat dikirim tepat waktu dan dapat menjawab semua pertanyaan bisnis yang penting
b. Pilihan terbaik untuk *data mart* yang pertama adalah yang berhubungan dengan sales, misal *property sales*, *property leasing*, *property advertising*.

Langkah 2 : Pemilihan sumber

a. Proses pemilihan secara pasti apa yang diwakili atau direpresentasikan oleh sebuah tabel fakta.

b. Misal, jika sumber dari sebuah tabel fakta *properti sale* adalah *properti sale* individual maka sumber dari sebuah dimensi pelanggan berisi rincian pelanggan yang membeli properti utama

Langkah 3 : Mengidentifikasi dimensi

a. Set dimensi yang dibangun dengan baik, memberikan kemudahan untuk memahami dan menggunakan *data mart*
b. Dimensi ini penting untuk menggambarkan fakta-fakta yang terdapat pada tabel fakta
c. Misal, setiap data pelanggan pada tabel dimensi pembeli dilengkapi dengan id_pelanggan, no_pelanggan, tipe_pelanggan, tempat_tinggal, dan lain sebagainya.
d. Jika ada dimensi yang muncul pada dua data mart,kedua *data mart* tersebut harus berdimensi sama, atau paling tidak salah satunya berupa subset matematis dari yang lainnya.
e. Jika sebuah dimensi digunakan pada dua *data mart* atau lebih,dan dimensi ini tidak disinkronisasi,maka keseluruhan *data warehouse* akan gagal, karena dua *data mart* tidak bisa digunakan secara bersama-sama

Langkah 4 : Pemilihan fakta

a. Sumber dari sebuah tabel fakta menentukan fakta mana yang bisa digunakan dalam *data mart*.
b. Semua fakta harus diekspresikan pada tingkat yang telah ditentukan oleh sumber

Langkah 5 : Menyimpan pre-kalkulasi di tabel fakta

a. Hal ini terjadi apabila fakta kehilangan statement

Langkah 6 : Melengkapi tabel dimensi

a. Pada tahap ini kita menambahkan keterangan selengkap - lengkapnya pada tabel dimensi
b. Keterangannya harus bersifat *intuitif* dan mudah dipahami oleh pengguna

Langkah 7 : Pemilihan durasi database

a. Misalnya pada suatu perusahaan asuransi, mengharuskan data disimpan selama 10 tahun atau lebih

Langkah 8 : Menelusuri perubahan dimensi yang perlahan

a. Atribut dimensi yang telah berubah tertulis ulang
b. Atribut dimensi yang telah berubah menimbulkan sebuah dimensi baru
c. Atribut dimensi yang telah berubah menimbulkan alternatif

sehingga nilai atribut lama dan yang baru dapat diakses secara bersama pada dimensi yang sama.

Langkah 9 : Menentukan prioritas dan mode query

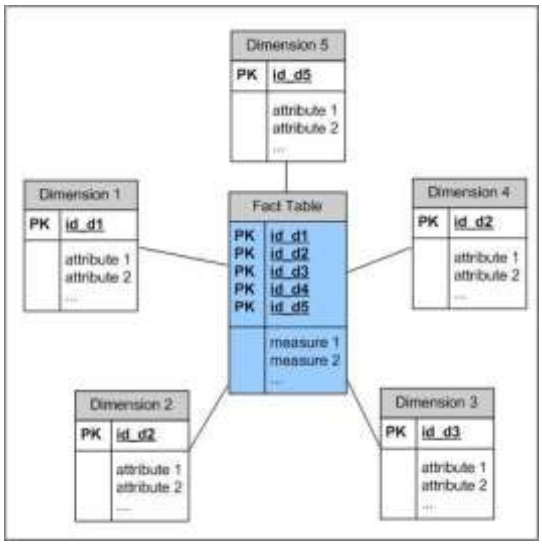
a. Pada tahap ini lebih menggunakan perancangan fisik.

2.2.4.7 Skema *Data Warehouse*

Skema yang sering digunakan pada *data warehouse* adalah skema bintang atau *snowflake*, kedua skema ini sangat mudah dimengerti dan sesuai dengan kebutuhan bisnis, mendukung *query* sederhana dan menyediakan performa *query* yang superior dengan meminimalisasi tabel-tabel join **[7]**.

1. Skema Bintang (*Star Schema*)

Skema bintang merupakan struktuk logikal yang memiliki tabel fakta yang terdiri atas data faktual ditengahnya, dan dikelilingi oleh tabel-tabel dimensi yang berisi referensi data.

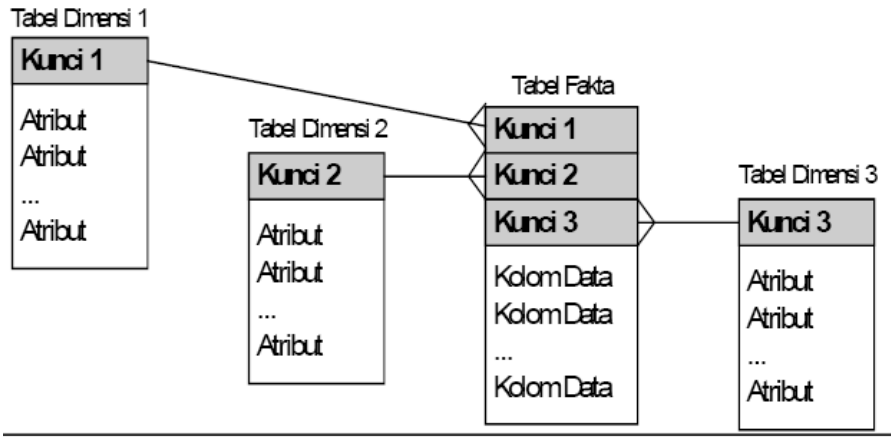


Gambar 2.9 Skema Bintang

Jenis – jenis Skema Bintang antara lain :

a. Skema bintang sederhana

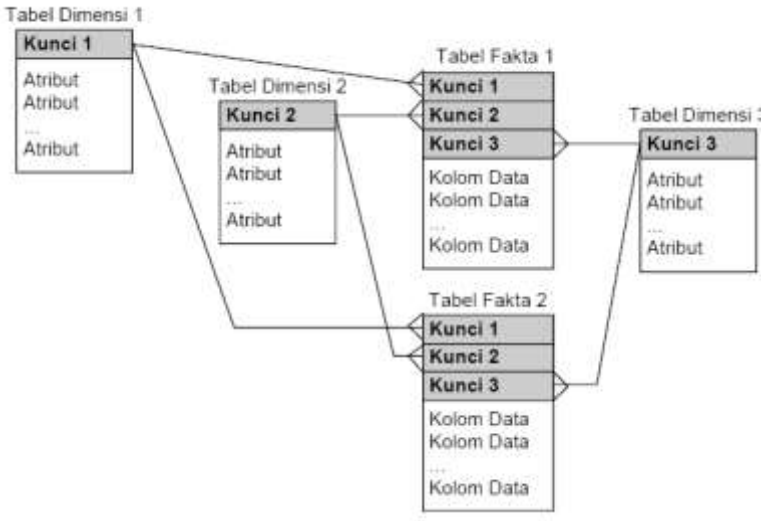
Dalam skema ini, setiap table harus memiliki *primary key* yang terdiri dari satu kolom atau lebih. *Primary key* dari table fakta terdiri dari satu atau lebih *foreign key*. *Foreign key* merupakan *primary key* pada table lain.



Gambar 2.10 Skema Bintang Sederhana

b. Skema bintang dengan banyak table fakta

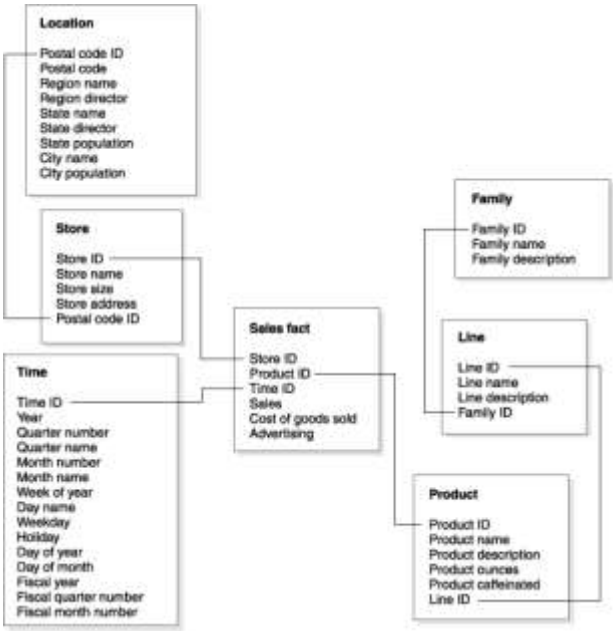
Skema bintang juga bisa terdiri dari satu atau lebih table fakta. Dikarenakan karena table fakta tersebut ada banyak, misalnya disamping penjualan terdapat table fakta *forecasting* dan *result*. Walaupun terdapat lebih dari satu table fakta, mereka tetap menggunakan table dimensi bersama-sama.



Gambar 2.11 Skema Bintang Dengan Banyak Tabel Fakta

2. *Snowflake Schema*

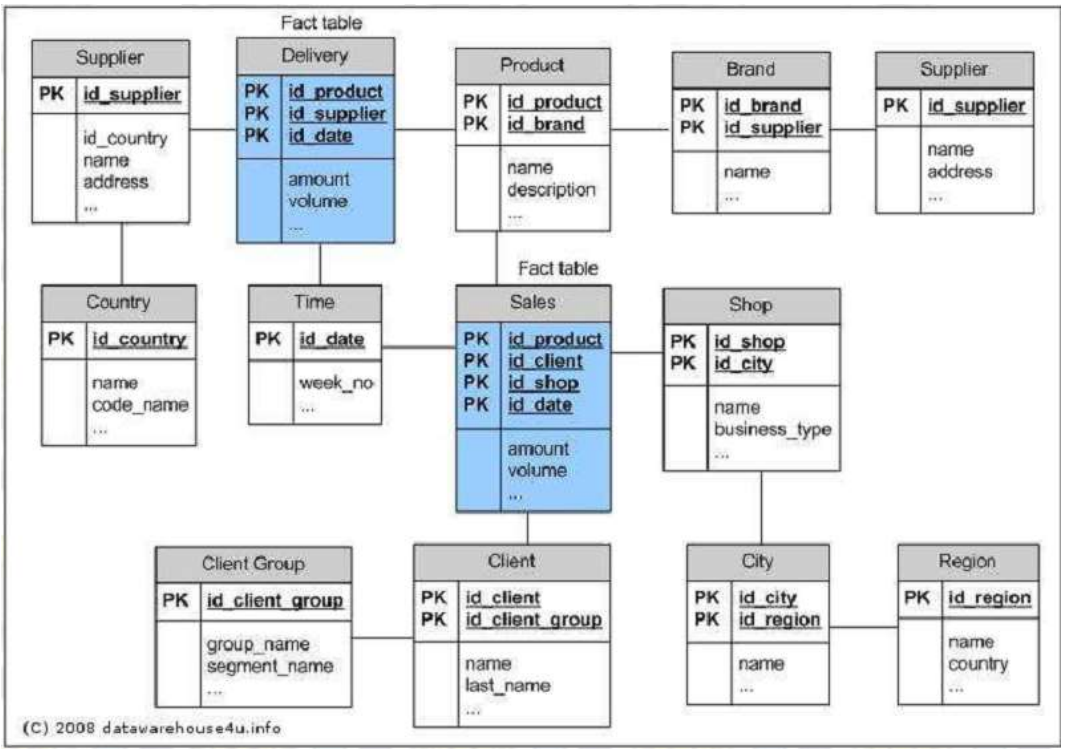
Merupakan varian dari skema bintang dimana table-table dimensi tidak terdapat data yang di denormalisasi. Dengan kata lain satu atau lebih table dimensi tidak bergabung secara langsung kepada table fakta tapi pada table dimensi lainnya.



Gambar 2.12 *Snowflake Schema*

3. *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 2.13 *Fact Constellation Schema*

2.2.4.8 Tujuan *Data Warehouse*

Data warehouse biasanya dimiliki oleh perusahaan yang sudah besar, yang memiliki banyak cabang, data yang banyak dan tentunya struktur organisasi yang kompleks **[3]**.

Perusahaan yang memiliki banyak transaksi, yang memiliki banyak cabang, yang mengakibatkan data-data barang perusahaan tersebar dilokasi yang berbeda, system operasi yang berbeda, bahkan di basisdata yang berbeda. Sehingga dalam pengambilan sebuah keputusan pihak perusahaan membutuhkan akses ke semua sumber data pada tiap-tiap cabang dan ini tidak efisien. Dari permasalahan ini, *Data warehouse* hadir sebagai solusi yang kompetitif, efisien untuk pengambilan keputusan di perusahaan.

Tujuan Akhir menggunakan Datawarehouse

1. Menyediakan data organisasi yang mudah diakses oleh manager.
2. Data yang berada di *data warehouse* bersifat konsisten, dan merupakan kebenaran.

3. *Data warehouse* merupakan tempat, dimana data yang telah digunakan di publikasikan.
4. Kualitas data di *data warehouse* dapat diandalkan.

2.3 Alat-alat Pemodelan Sistem

Alat-alat pemodelan sistem membahas mengenai *tools* apa saja yang digunakan dalam membuat tugas akhir ini.

2.3.1 Diagram Konteks

diagram konteks merupakan diagram yang menggambarkan kondisi sistem yang ada baik input maupun output serta menyertakan terminator yang terlibat dalam penggunaan sistem. Diagram ini akan memberi gambaran tentang keseluruhan sistem. Sistem dibatasi oleh *boundary* (dapat digambarkan dengan garis putus). Dalam diagram konteks hanya ada satu proses. Tidak boleh ada store dalam diagram konteks **[8]**.

Diagram konteks berisi gambaran umum (secara garis besar) sistem yang akan dibuat. Secara kalimat, dapat dikatakan bahwa diagram konteks ini berisi "siapa saja yang memberi data (dan data apa saja) ke sistem, serta kepada siapa saja informasi (dan informasi apa saja) yang harus dihasilkan sistem." Maka dapat disimpulkan bahwa diagram konteks adalah diagram yang terdiri dari suatu proses dan menggambarkan ruang lingkup suatu sistem. Diagram konteks merupakan level tertinggi dari DFD yang menggambarkan seluruh *input* ke sistem atau *output* dari sistem.

2.3.2 *Data Flow Diagram* (DFD)

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 sering disebut juga dengan nama *Bubble chart, Bubble diagram, model proses*, diagram alur kerja, atau model fungsi. 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 **[8]**.

DFD 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.

2.3.3 *Entity Relation Diagram* (ERD)

Entity Relationship Diagram merupakan model data berupa notasi grafis dalam pemodelan data konseptual yang menggambarkan hubungan antara penyimpan satu dengan penyimpan yang lain. Model data sendiri merupakan sekumpulan cara, peralatan untuk mendeskripsikan data-data yang memiliki hubungan satu sama lain, semantiknya, serta batasan konsistensi. Model data terdiri dari model hubungan entitas dan model relasional. Diagram hubungan entitas ditemukan oleh Peter Chen dalam buku *Entity Relational Model-Toward a Unified of Data*. Pada saat itu diagram hubungan entitas dibuat sebagai bagian dari perangkat lunak yang juga merupakan modifikasi khusus, karena tidak ada bentuk tunggal dan standar dari diagram hubungan entitas **[8]**.

2.4 Alat-alat Pembangunan Perangkat Lunak

Alat-alat Pembangunan Perangkat Lunak membahas mengenai alat atau tools apa saja yang digunakan dalam membuat Aplikasi *data warehouse* dalam penelitian ini.

2.4.1 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 **[9]**:

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.
5) 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.

2.4.2 *SQL Server*

SQL Server adalah sistem manajemen database relasional (RDBMS) yang dirancang untuk aplikasi dengan arsitektur *client/server*. Istilah *client, server,* dan *client/server* dapat digunakan untuk merujuk kepada konsep yang sangat umum atau hal yang spesifik dari perangkat keras atau perangkat lunak. Pada level yang sangat umum, sebuah *client* adalah setiap komponen dari sebuah sistem yang meminta layanan atau sumber daya (*resource*) dari komponen sistem lainnya. Sedangkan sebuah *server* adaah setiap komponen sistem yang menyediakan layanan atau sumber daya ke komponen sistem lainnya. **[8]**.