BAB II
TINJAUAN PUSTAKA

2.1 Profil Tempat Kerja Praktek

CV. Perkasa adalah Perusahaan Perorangan yang bergerak di bidang jasa. Perusahaan ini merupakan salah satu mitra Telkom yang bergerak di bidang penjualan salah satu prodak Telkom yaitu Telkom speedy. Dimana CV. Perkasa ini menjadi pihak ketiga diantara pelanggan dan PT Telkom dalam pelayanan pemasangan internet Speedy.

2.1.1 Visi CV. Perkasa:

Untuk menjadi mitra terbaik PT. Telkom Indonesia.

2.1.2 Misi CV. Perkasa:

1. Memberikan pelayanan yang dapat memuaskan konsumen.
2. Menjadi model peran sebagai pusat agensi terkemuka.

2.1.3 Sejarah Instansi

CV. Perkasa memiliki visi menjadi salah satu CV terbaik dalam bidang jasa khususnya pemasaran, untuk mencapai visi tersebut CV. Perkasa menjalankan misi nya dengan selalu memberikan layanan yang terbaik dan memuaskan bagi pelanggan.

Berdirinya CV. Perkasa diawali dengan sebuah ide dari tiga orang yakni Pak Heri Suseno, Pak Dadang, dan Pak Arliyawan. Yang mana Pak Heri Suseno awalnya bekerja di Kopegtel (Koperasi Pegawai Telkom) sampai masa habis kerja (pensiun), setelah itu dia ditawari oleh temannya yang bekerja di Telkom Sepeedy untuk membangun sebuah Agency dimana agensi ini nantinya akan menjadi Mitra Telkom dalam membantu penjualan produk Telkom yaitu Internet Speedy. Pada saat itulah Pak Heri Suseno bertemu dengan Pak Dadang dan Pak Arliyawan yang pada saat itu mereka bekerja sebagai SF (*Sales Force*) di sebuah agensi Telkom juga yaitu agency Triva. Lalu mereka mendirikan sebuah kantor di Jalan

Bengawan No 59. Dengan struktur kepengurusan Pak Heri Suseno sebagai Owner, Pak Dadang sebagai General Manajer dan Pak Arliyawan menjadi Pimpinan Cabang di Rancaekek.

Pada awal nya CV. Perkasa hanya mempunyai dua wilayah untuk pemasaran Speedy yaitu di daerah Rancaekek yang berkantor di STO Telkom Rancaekek dan di daerah Turangga yang berkantor di STO Telkom Turangga. Yang mana kantor yang berada di Jalan Bengawan digunakan sebagai kantor pusat. Seiring waktu berjalan kini kemajuan yang dicapai pun sangat pesat, puncaknya agensi ini pernah menjadi agensi nomor satu di Bandung dengan penjualan terbanyak se Indonesia. Sebagai penghargaan dari Telkom, agensi ini pun diberi kepercayaan untuk memperluas wilayah pemasaran Speedy hingga ke lima wilayah yaitu Turangga, Hegarmanah, Cicilengka, Centreum dan Padalarang. Dan hingga saat ini pun CV. Perkasa masih menjadi pusat pemasaran.

2.1.4 Logo Instansi

Deskripsi dari logo instansi CV. Perkasa yaitu, warna biru pada logo melambangkan bahwa CV. Perkasa merupakan salah satu mitra PT. Telkom yang dimana logo PT. Telkom didominasi oleh warna biru, dan gambar P pada logo melambangkan huruf pertama dari kata “PERKASA”.



Gambar 2.1 Logo CV. PERKASA

2.1.5 Badan Hukum Instansi

CV. Perkasa tidak memilik badan hukum dikarenakan prosedur CV itu sendiri, CV atau *Comanditaire Venootschap* adalah bentuk usaha

yang merupakan salah satu alternatif yang dapat dipilih oleh para pengusaha yang ingin melakukan kegiatan usaha dengan modal yang terbatas.

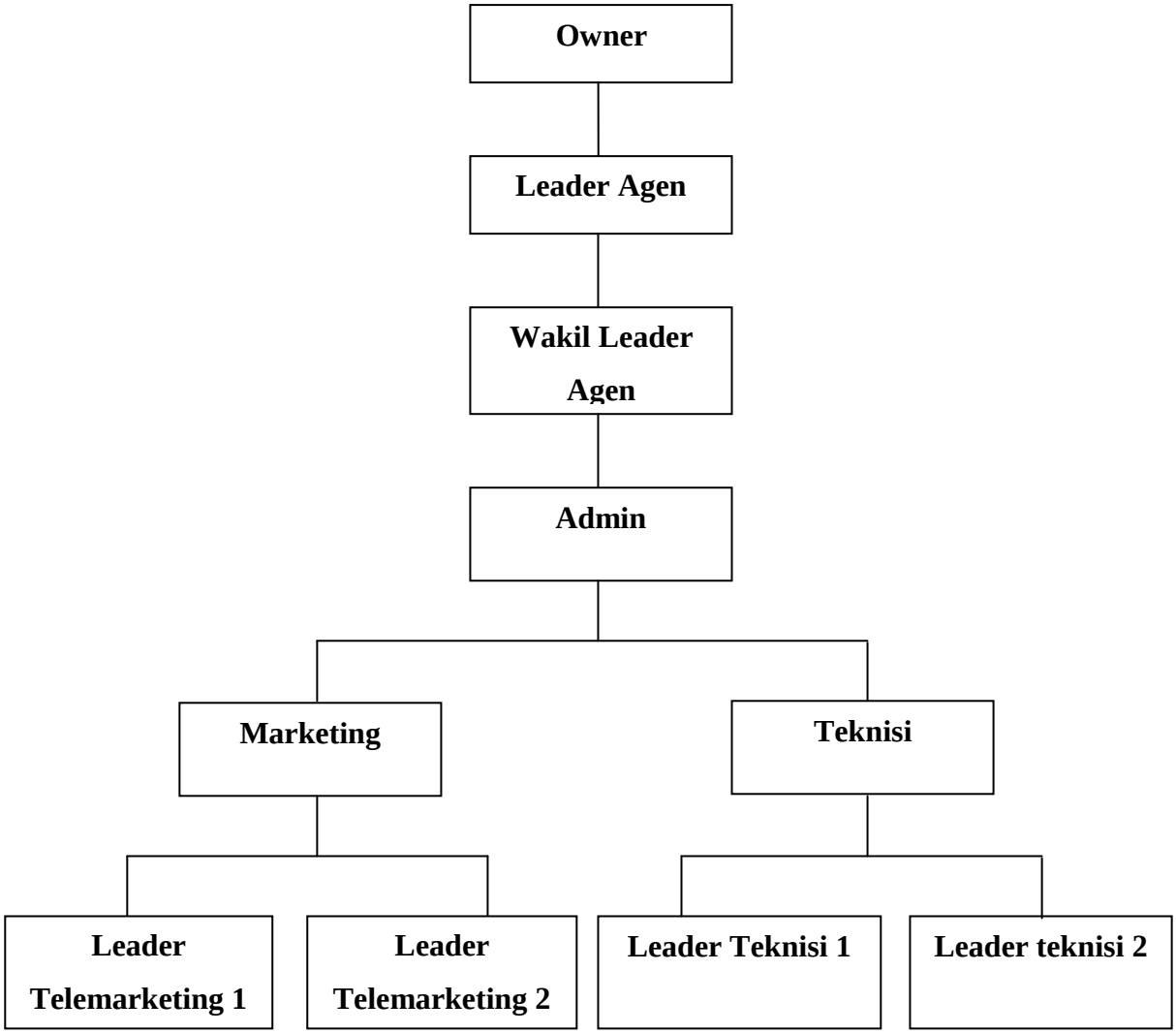
Karakteristik CV yang tidak dimiliki Badan Usaha lainnya adalah: CV didirikan minimal oleh dua orang, dimana salah satunya akan bertindak selaku Persero Aktif (persero pengurus) yang nantinya akan bergelar Direktur, sedangkan yang lain akan bertindak selaku Persero Komanditer (Persero diam). Seorang persero aktif akan bertindak melakukan segala tindakan pengurusan atas Perseroan; dengan demikian, dalam hal terjadi kerugian maka Persero Aktif akan bertanggung jawab secara penuh dengan seluruh harta pribadinya untuk mengganti kerugian yang dituntut oleh pihak ketiga. Sedangkan untuk Persero Komanditer, karena dia hanya bertindak selaku sleeping partner, maka dia hanya bertanggung jawab sebesar modal yang disetorkannya ke dalam perseroan.

2.1.6 Struktur Organisasi dan *Job Description*

CV. Perkasa merupakan salah satu agency dari perusahaan PT. Telkom, PT. Telkom sendiri merupakan salah satu perusahaan BUMN (Badan Usaha Milik Negara). Yang di pimpin oleh seorang general manager yang di pilih dan di tetapkan oleh *owner* (pemilik saham terbesar). Dalam struktur organisasinya nya cv perkasa terdiri dari beberapa jabatan dimana general manager menjadi penentu keputusan di perusahaan dan di bawahnya terdapat kepala STO (sentra telepon otomatis) dimana kepala STO ini berkewajiban mengatur para SF(sales force) dan tele marketing selain itu tugas utama kepala STO adalah menyusun laporan harian kepada general manager. *Applicator* bertugas untuk mengambil aplikasi atau persyaratan pendaftaran speedy dari pelanggan. Nantinya persyaratan pelanggan akan di limpahkan ke kantor Telkom speedy, selanjutnya persyaratan tersebut akan di proses oleh pihak Telkom.

CV. Perkasa bergerak di bidang marketing. dalam hal ini CV. Perkasa menjadi salah satu mitra dari PT. Telkom dalam penjualan salah

satu produk Telkom yaitu speedy. Speedy adalah salah satu produk Telkom, speedy merupakan provider internet yang hanya bisa di pasang jika ada sambungan telepon.



Gambar 1.2 Struktur Organisasi

2.1.7 Deskripsi Jabatan

1. Owner

Owner atau pemilik perusahaan yang bertugas sebagai penyedia biaya perencanaan dan pelaksanaan proyek, selain itu juga *owner* bertanggung jawab penuh atas perusahaannya.

2. Leader Agen

   Bertanggung jawab terhadap kegiatan karyawan CV.PERKASA, agar dapat memberikan pelayanan terbaik kepada calon pembeli atau konsumen.

3. Wakil Leader Agen

   Orang kedua setelah leader agen yang menggantikan posisi leader agen jika leader agen sedang meeting atau tugas keluar kota.

4. Admin

   Bertugas sebagai bagian yang mengatur pemasukan dan pengeluaran perusahaan dan mencatat penjualan perusahaan.

5. Leader Marketing 1

   Bekerja sebagai leader yang memimpin dan mengawasi karyawan bagian marketing, yang meliputi daerah Turangga, Kopo, dan Tegalega.

6. Leader Marketing 2

   Bekerja sebagai leader yang memimpin dan mengawasi karyawan bagian marketing, yang meliputi daerah Hegarmanah, Sukajadi, Setiabudi, Cihampelas dan Pasteur.

7. Leader Teknisi 1

   Bekerja sebagai kepala teknisi yang menyangkut semua hal tentang pemasangan speedy, yang meliputi daerah Turangga, Kopo dan Tegalega

8. Leader Teknisi 2

   Bekerja sebagai kepala teknisi yang menyangkut semua hal tentang pemasangan speedy, yang meliputi daerah Hegarmanah, Sukajadi, Setiabudi, Cihampelas dan Pasteur.

2.2 Landasan Teori

Beberapa landasan teori yang digunakan dalam pembangunan sistem aplikasi admin. Diantaranya adalah:

2.2.1 Basis Data

Basis Data adalah kumpulan informasi yang disimpan didalam komputer secara sistematik sehingga dapat diperiksa menggunakan suatu program komputer unutk memperoleh informasi dari basis data tersebut.

2.2.1.1 Sistem Basis Data

Sistem adalah sebuah tatanan (keterpaduan) yang terdiri atas sejumlah komponen fungsional (dengan satuan fungsi / tugas khusus) yang saling berhubungan dan secara bersama – sama bertujuan untuk memenuhi suatu proses atau pekerjaan tertentu. Basis data hanyalah sebuah objek yang pasif atau mati. Ia ada karena ada pembuatnya. Ia tidak akan pernah berguna jika tidak ada pengelola atau penggeraknya. Yang menjadi penggeraknya secara langsung adalah program atau aplikasi (*software*). Gabungan keduanya menghasilakn sebuah sistem. Karena itu, secara umum sebuah sistem basis data merupakan sistem yang terdiri atas kumpulan file (table) yang saling berhubungan (dalam sebuah basis data di sebuah sistem komputer) dan sekumpulan program (DBMS) yang memungkinkan beberapa pemakai dan atau program lain untuk mengakses dan memanipulasi file – file (table - tabel) tersebut.

Lebih jauh lagi, dalam sebuah sistem basis data, secara lengkap akan terdapat komponen – komponen utama sebagai berikut :

a. Perangkat keras (*Hardware*)

Perangkat keras yang biasanya terdapat dalam sebuah sistem basis data adalah :

1. Komputer (satu untuk sistem yang *stand – alone* atau lebih dari satu untuk sistem jaringan).
2. Memori sekunder yang *on – line* (*harddisk*).
3. Memori sekunder yang *off – line* (*Tape* atau *Removable Disk*) untuk keperluan *backup* data.

4. Media atau perangkat komunikasi (untuk sistem jaringan).

b. Sistem Operasi (*Operating System*)

Secara sederhana, sistem operasi merupakan program yang mengaktifkan atau memfungsikan sistem komputer, mengendalikan seluruh sumber daya (*resource*) dalam komputer dan melakukan operasi – operasi dasar dalam computer (operasi I/O, pengelolaan file, dll). Program pengelola basis data hanya dapat aktif (*running*) jika sistem operasi yang dikehendakinya atau sesuai telah aktif.

c. Basis Data (*Database*)

Sebuah sistem basis data dapat memiliki beberapa basis data. Setiap basis data dapat berisi sejumlah objek basis data seperti *file / table*, indeks, dll. Di samping berisi atau menyimpan data, setiap basis data juga mengandung definisi struktur (baik untuk basis data maupun objek – objeknya secara detail).

d. Sistem (Aplikasi / perangkat lunak) Pengelola Basis Data (DBMS)

Pengelolaan basis data secara fisik tidak dilakukan oleh pemakai secara langsung, tetapi ditangani oleh sebuah perangkat lunak yang khusus. Perangkat lunak ini disebut DBMS (*Database Management Sistem*) yang yang akan menentukan bagaimana data diorganisasi, disimpan, diubah dan diambil kembali. Ia juga menerapkan mekanisme pengamanan data, pemakaian data secara bersama, pemaksaan keakuratan atau konsistensi data, dan sebagainya.

e. Pemakai (*User*)

Ada beberapa jenis pemakai terhadap suatu sistem basis data yang dibedakan berdasarkan cara mereka berinteraksi terhadap sistem :

1. Programmer Aplikasi

Pemakai yang berinteraksi dengan basis data melalui *Data Manipulation Language* (DML), yang disertakan (*embedded*) dalam program yang ditulis dalam bahasa pemrograman induk (seperti C, Pascal, Cobol, dan lain lain).

2. User Mahir (*Casual User*)

   Pemakai yang berinteraksi dengan sistem tanpa menulis modul program. Mereka menyatakan *query* (untuk akses data) dengan bahasa *query* yang disediakan oleh suatu DBMS.

3. User Umum (*End User / Naïve User*)

   Pemakai yang berinteraksi dengan sistem basis data melalui pemanggilan satu program aplikasi permanen (*executable program*) yang telah ditulis atau disediakan sebelumnya.

4. User Khusus (*Specialized User*)

   Pemakai yang menulis aplikasi basis data non-konvensional, tetapi untuk keperluan – keperluan khusus, seperti untuk aplikasi AI, Sistem Pakar, Pengolahan Citra, dll, yang bisa saja mengakses basis data dengan / tanpa DBMS yang bersangkutan.

f. Aplikasi (perangkat lunak) lain (bersifat opsional)

Aplikasi (perangkat lunak) lain ini bersifat opsional. Artinya, ada atau tidaknya tergantung pada kebutuhan kita. DBMS yang kita gunakan lebih berperan salam pengorganisasian data dalam basis data, sementara bagi pemakai basis data (khususnya yang menjadi *end – user* / *naive - user*) dapat dibutuhkan / disediakan program khusus / lain untuk melakukan pengisian, pengubahan dan pengambilan data. Pemrogram ini ada yang sudah disediakan bersama dengan DBMS – nya, ada juga yang harus dibuat sendiri dengan menggunakan aplikasi lain yang khusus untuk itu (*delevopment tools*).

2.2.1.2 Model Data

Model data sendiri dapat didefinisikan sebagai kumpulan perangkat konseptual untuk menggambarkan data, hubungan data, semantic (makna) data dan batasan data. Oleh karena yang ingin ditunjukkan adalah makna dari data dan keterhubungannya dengan data

lain, maka Model Data ini lebih tepat jika disebut Model Data Logik. Penyingkatannya menjadi Model Data dilakukan demi penyederhanaan penulisan.

Ada sejumlah secara dalam meprestasikan Model Data dalam perancangan basis data, yang secara umum dapat dibagi dalam 2 (dua) kelompok yaitu, yaitu :

a. Model Logik Data Berdasarkan Objek (*Objek – Based Logical Models*) yang terdiri dari :
    1. *Entity – Relationship Model*
    2. *Object – Oriented Model*
    3. *Semantic Data model*
    4. *Functional Data model*
b. Model Lojik Data Berdasarkan *Record* (*Record – Based Locical* Models) yang terdiri dari :
    1. *Relational Model*
    2. *Hierarchical Model*
    3. *Network Model*

2.2.1.2.1 ERD

ERD merupakan suatu model untuk menjelaskan hubungan antar data dalam basis data berdasarkan objek-objek dasar data yang mempunyai hubungan antar relasi.

a. Entiti

Entiti merupakan objek yang mewakili sesuatu yang nyata dan dapat dibedakan dari sesuatu yang lain (Fathansyah, 1999: 30). Simbol dari entiti ini biasanya digambarkan dengan persegi panjang.

b. Atribut

Setiap entitas pasti mempunyai elemen yang disebut atribut yang berfungsi untuk mendeskripsikan karakteristik dari entitas tersebut. Isi dari atribut mempunyai sesuatu yang dapat mengidentifikasikan isi elemen satu dengan yang lain. Gambar atribut diwakili oleh simbol elips.

c. Hubungan / Relasi

Hubungan antara sejumlah entitas yang berasal dari himpunan entitas yang berbeda. Relasi dapat digambarkan sebagai berikut :

Relasi yang terjadi diantara dua himpunan entitas (misalnya A dan B) dalam satu basis data yaitu (Abdul Kadir, 2002: 48) :

1. Satu ke satu (*One to one*)

Hubungan relasi satu ke satu yaitu setiap entitas pada himpunan entitas A berhubungan paling banyak dengan satu entitas pada himpunan entitas B.

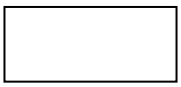
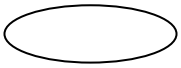
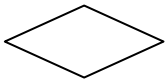
2. Satu ke banyak (*One to many*)

Setiap entitas pada himpunan entitas A dapat berhubungan dengan banyak entitas pada himpunan entitas B, tetapi setiap entitas pada entitas B dapat berhubungan dengan satu entitas pada himpunan entitas A.

3. Banyak ke banyak (*Many to many*)

Setiap entitas pada himpunan entitas A dapat berhubungan dengan banyak entitas pada himpunan entitas B.

Table 2.1 Notasi – Notasi Simbolik ERD (Entity Relationship Diagram)

SIMBOL	ARTI
	Persegi panjang, ,menyatakan Himpunan Entitas.
	Elips menyatakan Atribut (Atribut yang berfungsi sebagai *key* digarisbawahi).
	Belah ketupat menyatakan Himpun Relasi.
	Garis sebagai penghubung antara Himpunan Relasi dengan Himpunana Entitas dan Himpunan Entitas dengan atributnya.
a. 1 dan 1 untuk relasi satu-ke-satu.	Kardinalitas Relasi dapat dinyatakan dengan banyaknya garis cabang atau dengan pemakaian angka.

b. 1 ke N untuk relasi satu-ke-banyak. c. N ke N untuk relasi banyak-ke-banyak.	

2.2.1.2.2 Varian Entitas dan Relasi

Idealnya himpunan entitas yang terlibat dalam sebuah ERD adalah himpunan entitas kuat / bebas. Himpunan entitas demikian tidak memiliki ketergantungan dengan himpunan entitas lainnya. Namun demikian, dalam pembentukan ERD kita tidak selalu dapat melibatkan himpunan entitas seperti itu. Ada kalanya kita juga melibatkan himpunan entitas yang lemah atau merupakan bagian dari himpunan entitas lainnya.

2.2.1.2.2.1 Himpunan Entitas Lemah (*Weak Entity Sets*)

Berisi entitas – entitas yang kemunculannya tergantung pada eksistensinya dalam sebuah relasi terhadap entitas lain. Himpunan entitas demikian biasanya tidak memiliki atribut yang dapat berfungsi sebagai *key*, yang dapat menjamin keunikan entitas di dalamnya.

2.2.1.2.2.2 Sub Entitas (*Subtype Entities*)

Merupakan himpunan entitas yang beranggotakan entitas – entitas yang merupakan bagian dari himpunan entitas yang lebih *superior* (utama). Sub Entitas ini merupakan hasil dekomposisi (spesialisasi) himpunan entitas berdasarkan pengelompokan tertentu.

2.2.1.3 Arsitektur Sistem

Arsitektur sistem merujuk pada konfigurasi sistem secara keseluruhan yang akan menjadi “tempat hidup” dari DBMS, basis data dan aplikasi yang memanfaatkannya. Sudah seharusnya ditetapkan sejak awal sebelum memulai perancangan basis data, atau paling tidak sebelum melaksanakan tahap implementasi basis data.

Beberapa jenis arsitektur sistem yang dapat digunakan dan akan diuraikan pada bab ini adalah :

a. Sistem Tunggal (*Stand - Alone*).
b. Sistem Tersentralisasi (*Centralized Sistem*).
c. Sistem *Client - Server*

2.2.1.3.1 Sistem Tunggal / Mandiri (*Stand-Alone*)

Pada arsitektur ini, DBMS, basis data dan aplikasi basis data ditempatkan pada computer yang sama dengan demikian, pemakai yang dapat menggunakannya disetiap saat juga hanya satu orang (*single user*). Arsitektur ini merupakan arsitektur sistem yang paling sederhana dan paling murah dan dapat menjadi sebuah pilihan jika basis data yang dikelola tidak terlalu besar dan lebih bersifat membantu mempercepat pekerjaan administrative.

2.2.1.3.2 Sistem Tersentralisasi (*Centralized Sistem*)

Arsitektur ini terdiri atas sebuah mesin server dan sejumlah terminal (yang menjadi tempat *user* berinterkasi dengan sistem). Yang tersentralisasi dalam arsitektur ini dapat mencakup basis data, DBMS dan aplikasi basis data atau basis data saja. Untuk lingkup sentralisasi yang pertama, maka jenis *server*-nya sering disebut sebagai DBMS atau server aplikasi dan terminalnya lebih tepat disebut *dumb-terminal* (terminal pasif). Sedangkan jika yang disentralisasi hanya basis data, server yang digunakan bias disebut *file-server* dan terminalnya disebut *work station*.

2.2.2 DBMS (*DataBase Management System*)

DBMS (*DataBase Management System***)** adalah sistem yang secara khusus dibuat untuk memudahkan pemakai dalam mengelola basis data. Sistem ini dibuat untuk mengatasi kelemahan sistem pemrosesan yang berbasis berkas.Pada pendekatan yang berbasis berkas, umumnya perancangan sistem didasarkan pada kebutuhan individual pemakai, bukan

berdasarkan kebutuhan sejumlah pemakai. Setiap kali terdapat kebutuhan baru dari seorang pemakai, kebutuhan segera diterjemahkan kedalam program komputer. Akibatnya, kemungkinan besar setiap program aplikasi menuliskan data tersendiri. Sementara itu ada kemungkinan data yang sama juga terdapat pada berkas-berkas lain yang digunakan oleh program aplikasi lain.

DBMS merupakan perantara bagi pemakai dengan basis data. Untuk merinteraksi dengan DBMS (basis data) menggunakan bahasa basis data yang telah ditentukan oleh perusahaan DBMS. Bahasa basis data biasanya terdiri atas perintah-perintah yang di formulasikan sehingga perintah tersebut akan diproses olah DBMS. Perintah-perintah biasanya ditentukan oleh user. Ada 2 bahasa basis data:

2.2.2.1 Data Definition Language (DDL)

DDL digunakan untuk menggambarkan desain basis data secara keseluruhan. DDL digunakan untuk membuat tabel baru, menuat indeks, ataupun mengubah tabel. Hasil kompilasi DDL disimpan di kamus data. Contoh dari DDL yaitu pembuatan database memakai sql

```
CREATE TABLE IF NOT EXISTS `kost` (
 `Nama` varchar(50) NOT NULL,
 `Lokasi` varchar(50) NOT NULL,
 `Fasilitas` varchar(50) NOT NULL,
 `Alat_transportasi` varchar(30) NOT NULL,
 `Status_penyewaan` varchar(20) NOT NULL,
 `Harga` int(11) NOT NULL,
 `Foto_kost` blob NOT NULL,
```

```
  PRIMARY KEY (`Nama`)

) ENGINE=MyISAM DEFAULT CHARSET=latin1;
```

2.2.2.2 Data Manipulation Language (DML)

DML digunakan untuk melakukan menipulasi dan pengambilan data pada suatu basis data seperti penambahan data baru ke dalam basis data, menghapus data dari suatu basis data dan pengubahan data di suatu basis data. Dalam pembuatan DBMS diperlukan beberapa komponen fungsional penyusunnya sebagai berikut:

a. DML Precompiler yaitu mengkonversi pernyataan-pernyataan DML yang dimasukkan di dalam program aplikasi ke dalam pemanggilan prosedur normal di dalam bahasa induknya. *Procompiler* harus berinteraksi dengan *query processor* untuk membuat kode-kode yang diperlukan.
b. *Query Processor* yaitu menterjemahkan pernyataan-pernyataan bahasa *query* ke dalam instruksi-instruksi *low-level* yang dimengerti oleh *database manager*.
c. DDL *Compiler* yaitu mengkonversi pernyataan DDL ke dalam sekumpulan tabel yang mengandung metadata atau "data mengenai data".
d. Database Manager yaitu menyediakan *interface* antara data *low-level* yang disimpan didalam basisdata dengan program-program aplikasi dan queries yang dikirimkan ke sistem.

Salah satu tujuan dari DBMS adalah memberikan tampilan kepada pengguna dalam hal menyampaikan data. Untuk itu dalam DBMS terdapat Level Abstraksi Data. Level ini berguna untuk menyembunyikan detail atau kompleksitasnya basis data seperti bagaimana data disempan dan diolah. Sehingga pengguna hanya melihat tampilan yang dibutuhkan oleh pengguna. Contoh dari DML adalah *select* * pelanggan *where name* = 'unang'.

2.2.2.3 Level fisik

Level fisik merupakan level yang paling bawah. Pada level ini memperlihatkan bagaimana sesungguhnya data disimpan.

2.2.2.4 Level Konseptual

Level ini menggambarkan bagaimana sebenarnya basis dta disimpan dan berhubungan dengan data lainnya.

2.2.2.5 Level View

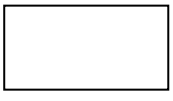
Level abstaraksi ini hanya menunjukkan sebagian dari basis data. Pada umumnya pengguna tidak melibatkan secara langsung sehingga pengguna hanya melihat data sesuai dengan yang dibutuhkan.

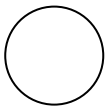
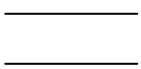
2.2.3 DFD (Data Flow Diagram)

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 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.

Table 2.2 Komponen Data Flow Diagram

KOMPONEN	ARTI
	Entitas Luar

	Proses
	Data Store
	Alur Data

2.2.3.1 Komponen Terminator / Entitas Luar

Terminator mewakili entitas eksternal yang berkomunikasi dengan sistem yang sedang dikembangkan. Biasanya terminator dikenal dengan nama entitas luar (*external entity*).

Komponen terminator ini perlu diberi nama sesuai dengan dunia luar yang berkomunikasi dengan sistem yang sedang dibuat modelnya, dan biasanya menggunakan kata benda, misalnya Admin.

2.2.3.2 Komponen Proses

Komponen proses menggambarkan bagian dari sistem yang mentransformasikan input menjadi output. Proses diberi nama untuk menjelaskan proses/kegiatan apa yang sedang/akan dilaksanakan. Pemberian nama proses dilakukan dengan menggunakan kata kerja *transitif* (kata kerja yang membutuhkan obyek).

2.2.3.3 Komponen Data Store

Komponen ini digunakan untuk membuat model sekumpulan paket data dan diberi nama dengan kata benda jamak, misalnya Mahasiswa.

Data store ini 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.

Suatu data store dihubungkan dengan alur data hanya pada komponen proses, tidak dengan komponen DFD lainnya. Alur data yang menghubungkan data store dengan suatu proses mempunyai pengertian sebagai berikut :

a. Alur data dari data store yang berarti sebagai pembacaan atau pengaksesan satu paket tunggal data, lebih dari satu paket data, sebagian dari satu paket tunggal data, atau sebagian dari lebih dari satu paket data untuk suatu proses
b. Alur data ke data store yang berarti sebagai pengupdatean data, seperti menambah satu paket data baru atau lebih, menghapus satu paket atau lebih, atau mengubah/memodifikasi satu paket data atau lebih .

2.2.3.4 Komponen Data Flow / Alur Data

Suatu data flow / alur data digambarkan dengan anak panah, yang menunjukkan arah menuju ke dan keluar dari suatu proses. Alur data ini digunakan untuk menerangkan perpindahan data atau paket data/informasi dari satu bagian sistem ke bagian lainnya. Alur data perlu diberi nama sesuai dengan data/informasi yang dimaksud, biasanya pemberian nama pada alur data dilakukan dengan menggunakan kata benda.

2.2.4 Desain Perangkat Lunak

Desain perangkat luank yaitu tahapan atau aktivitas yang difokuskan pada spesifikasi detil dari solusi berbasis komputer (*Whitten et al, 2004*).

Disain perangkat lunak sering juga disebut sebagai physical design. Jika tahapan analisis sistem menekankan pada masalah bisnis (*business rule*), maka sebaliknya disain perangkat lunak fokus pada sisi teknis dan implementasi sebuah perangkat lunak (*Whitten et al, 2004*).

Output utama dari tahapan desain perangkat lunak adalah spesifikasi disain. Spesifikasi ini meliputi spesifikasi desain umum yang akan disampaikan kepada *stakeholder* sistem dan spesifikasi desain rinci

yang akan digunakan pada tahap implementasi. Spesifikasi desain umum hanya berisi gambaran umum agar *stakeholder* sistem mengerti akan seperti apa perangkat lunak yang akan dibangun. Biasanya diagram USD tentang perangkat lunak yang baru merupakan point penting dibagian ini. Spesifikasi disain rinci atau kadang disebut desain arsitektur rinci perangkat lunak diperlukan untuk merancang sistem sehingga memiliki konstruksi yang baik, proses pengolahan data yang tepat dan akurat, bernilai, memiliki aspek *user friendly* dan memiliki dasar-dasar untuk pengembangan selanjutnya.

Desain arsitektur ini terdiri dari desain database, desain proses, desain user interface yang mencakup desain input, output form dan report, desain hardware, software dan jaringan. Desain proses merupakan kelanjutan dari pemodelan proses yang dilakukan pada tahapan analisis.

2.2.5 Pengujian (*Blackbox*)

Pengujian Perangkat Lunak Adalah elemen kritis dari jaminan kualitas perangkat lunak dan merepresentasikan kajian pokok dari spesifikasi, desain, dan pengkodean.

Pentingnya pengujian perangkat lunak dan implikasinya yang mengacu pada kualitas perangkat lunak tidak dapat terlalu ditekan karena melibatkan sederetan aktivitas produksi di mana peluang terjadinya kesalahan manusia sangat besar dan arena ketidak mampuan manusia untuk melakukan dan berkomunikasi dengan sempurna maka pengembangan perangkat lunak diiringi dengan aktivitas jaminan kualitas.

Meningkatnya visibilitas (kemampuan) perangkat lunak sebagai suatu elemen sistem dan “biaya” yang muncul akibat kegagalan perangkat lunak, memotivasi dilakukannya perencanaan yang baik melalui pengujian yang teliti. Pada dasarnya, pengujian merupakan satu langkah dalam proses rekayasa perangkat lunak yang dapat dianggap sebagai hal yang merusak dari pada membangun.

Sejumlah aturan yang berfungsi sebagai sasaran pengujian pada perangkat lunak adalah:

1. Pengujian adalah proses eksekusi suatu program dengan maksud menemukan kesalahan.
2. *Test case* yang baik adalah *test case* yang memiliki probabilitas tinggi untuk menemukan kesalahan yang belum pernah ditemukan sebelumnya.
3. Pengujian yang sukses adalah pengujian yang mengungkap semua kesalahan yang belum pernah ditemukan sebelumnya.

Sasaran itu berlawanan dengan pandangan yang biasanya dipegang yang menyatakan bahwa pengujian yang berhasil adalah pengujian yang tidak ada kesalahan yang ditemukan. Data yang dikumpulkan pada saat pengujian dilakukan memberikan indikasi yang baik mengenai reliabilitas perangkat lunak dan beberapa menunjukkan kualitas perangkat lunak secara keseluruhan, tetapi ada satu hal yang tidak dapat dilakukan oleh pengujian, yaitu "*pengujian tidak dapat memperlihatkan tidak adanya cacat, pengujian hanya dapat memperlihatkan bahwa ada kesalahan perangkat lunak".*

Sebelum mengaplikasikan metode untuk mendesain *test case* yang efektif, perekayasa perangkat lunak harus memahami prinsip dasar yang menuntun pengujian perangkat lunak, yaitu:

a. semua pengujian harus dapat ditelusuri sampai ke persyaratan pelanggan, maksudnya mengungkap kesalahan dari cacat yang menyebabkan program gagal.
b. Pengujian harus direncanakan lama sebelum pengujian itu mulai, maksudnya semua pengujian dapat direncanakan dan dirancang sebelum semua kode dijalankan.
c. Prinsip *Pareto* berlaku untuk pengujian perangkat lunak, maksudnya dari 80% kesalahan yang ditemukan selama pengujian dapat ditelusuri sampai 20% dari semua modul program.
d. Pengujian harus mulai "dari yang kecil" dan berkembang ke pengujian "yang besar", Selagi pengujian berlangsung maju, pengujian mengubah fokus dalam usaha menemukan kesalahan pada *cluster* modul yang terintegrasi dan akhirnya pada sistem.

e. Pengujian yang mendalam tidak mungkin karena tidak mungkin mengeksekusi setiap kombinasi jalur skema pengujian dikarenakan jumlah jalur permutasi untuk program menengah pun sangat besar.
f. Untuk menjadi paling efektif, pengujian harus dilakukan oleh pihak ketiga yang *independent*.

Dalam lingkungan yang ideal, perekayasa perangkat lunak mendesain suatu program komputer, sebuah sistem atau produk dengan *testabilitas* dalam pikirannya. Hal ini memungkinkan individu yang berurusan dengan pengujian mendesain *test case* yang efektif secara lebih mudah. *Testabilitas* adalah seberapa mudah sebuah program computer dapat diuji. Karena sangat sulit, perlu diketahui apa yang dapat dilakukan untuk membuatnya menjadi lebih mudah. *Procedural* dan menggunakannya sebagai pedoman untuk menetapkan basis set dari jalur eksekusi.

Sasaran utama desain *test case* adalah untuk mendapatkan serangkaian pengujian yang memiliki kemungkinan tertinggi di dalam pengungkapan kesalahan pada perangkat lunak.

Teknik pengujian *black-box* berfokus pada domain informasi dari perangkat lunak, dengan melakukan *test case* dengan menpartisi domain input dari suatu program dengan cara yang memberikan cakupan pengujian yang mendalam.

Metode pengujian *graph-based* mengeksplorasi hubungan antara dan tingkah laku objek – objek program. Partisi ekivalensi membagi domain input ke dalam kelas data yang mungkin untuk melakukan fungsi perangkat lunak tertentu. Analisis nilai batas memeriksaa kemampuan program untuk menangani data pada batas yang dapat diterima.

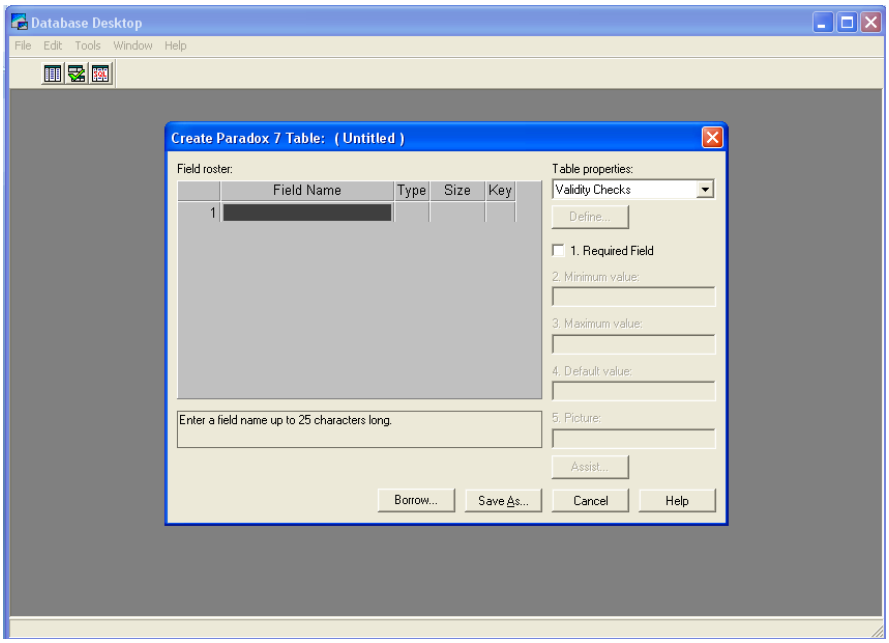
Metode pengujian yang terspesialisasi meliputi sejumlah luas kemampuan perangkat lunak dan area aplikasi. GUI, arsitektur *client/ server*, dokumentasi dan fasilitas *help* dan sistem *real time* masing – masing membutuhkan pedoman dan teknik khusus untuk pengujian perangkat lunak.

2.2.6 Query

Query adalah permintaan yang diberikan oleh user untuk mengambil informasi yang tersimpan dalam database.

2.2.7 Database Desktop

Database desktop adalah suatu *utility* yang disediakan untuk membuat tabel baru, membuat index, merubah struktur tabel yang telah dibuat bisa juga disebut resutruktur. Berikut tampilan dari Database Desktop:



Gambar 2.3 Tampilan Database Desktop

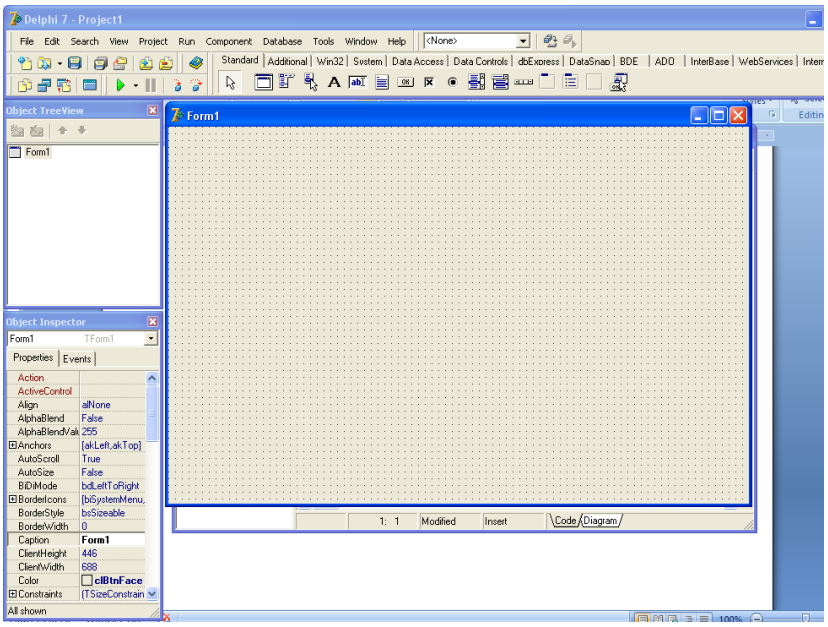
2.2.8 Borland Delphi 7

Borland Delphi 7.0 merupakan bahasa pemrograman berbasis *windows*. Borland Delphi 7.0 merupakan pilihan bagi sebagian *programmer* untuk membuat aplikasi, hai ini disebabkan kelebihan yang ada pada Borland Delphi 7.0 tersebut. Selain itu juga Delphi menggunakan bahasa pascal. Berikut ini sebagian kecil dari banyak kelebihan Borland Delphi 7.0 diantaranya :

a. Berbasiskan OOP (*Object Oriented Programming*). Setiap bagian yang ada pada program dipandang sebagai suatu object yang mempunyai sifat-sifat yang dapat diubah dan diatur.

b. IDE yang berkualitas. Delphi memiliki lingkungan pengembangan yang lengkap. Terdapat menu-menu yang memudahkan untuk mengatur proyek pengembangan *software*.

c. Delphi bersifat *multi-purpose*, dapat digunakan untuk berbagai keputusan pengembangan aplikasi mulai perhitungan sederhana sampai aplikasi multimedia bahkan yang terkoneksi ke internet. Tetapi harus dipahami bahwa kehebatan Delphi paling utama adalah cara termudah untuk mengakses Database menggunakan komponen-komponen yang disediakan.



Gambar 2.4 Tampilan aplikasi Delphi7