# 2. TEORI PENUNJANG

## 2.1. Internet

Pengertian akan istilah Internet pada laporan tugas akhir ini sangatlah penting mengingat sistem yang dibahas pada laporan ini dibuat dengan menggunakan fasilitas Internet. Internet merupakan singkatan dari *International Networking* yang berarti jaringan komputer dengan jangkuan internasional di mana komputer-komputer di berbagai tempat dari seluruh dunia terhubung satu sama lain dan berkomunikasi. Di berbagai tempat terdapat banyak jaringan lokal atau yang sering disebut *Local Area Network* (LAN) dan *Wide Area Network* (WAN). LAN dan WAN ini terhubung dan akhirnya saling membentuk suatu jaringan yang lebih luas yaitu internet tadi.

### 2.1.1. Sejarah Singkat Internet

Pada tahun 1969 *Defense Research Project Agency* (DARPA) membiayai suatu riset dan pembangunan proyek untuk membuat sebuah percobaan jaringan *packet switching*. Jaringan ini dibangun untuk meneliti teknik-teknik penyediaan komunikasi data yang dapat diandalkan dan bersifat *vendor-independent.*

Percobaan ARPANET berhasil sehingga banyak organisasi yang terhubung dengannya mulai menggunakannya untuk komunikasi sehari-hari. Pada tahun 1975, ARPANET dikonversikan dari sebuah jaringan percobaan menjadi jaringan operasi dan tanggung jawab untuk penataan jaringan tersebut diserahkan pada *Defense Communication Agency* (DCA). Bagaimana pun juga, pengembangan ARPANET tidak terhenti hanya karena digunakan sebagai jaringan operasi. Dasar dari protokol TCP/IP dikembangkan setelah ARPANET beroperasi.

Protokol TCP/IP diambil dari *Military Standards* (MIL STD) pada tahun 1983 dan semua *host* yang tehubung ke jaringan tersebut diharuskan untuk melakukan konversi ke protokol yang baru. Untuk memudahkan konversi ini DARPA membiayai Bolt, Bernaek, and Newman (BBN) untuk

mengimplementasikan TCP/IP di Berkeley (BSD) UNIX, Sehingga dimulailah penggabungan antara UNIX dan TCP/IP.

Kurang lebih pada saat TCP/IP diambil sebagai standard, istilah Internet digunakan lebih umum. Pada tahun 1983, ARPANET lama dibagi menjadi MILNET, bagian tak terklasifikasi dari *Defense Data Network* (DDN), dan sebuah ARPANET baru yang lebih kecil. Istilah internet biasanya merujuk ke keseluruhan jaringan: MILNET dan ARPANET. Pada tahun 1990, ARPANET secara resmi sudah tidak ada, tapi sekarang Internet sudah menjadi lebih besar bahkan melebihi semua jaringan di dunia.

### 2.1.2. TCP/IP

TCP/IP merupakan singkatan dari *Transmission Control Protocol/Internet* Protocol. Seperti disebutkan diatas TCP/IP merupakan tulang punggung Internet TCP/IP memiliki beberapa karakteristik yang sangat penting yang memenuhi kebutuhan akan komunikasi data sedunia sebagai berikut:

- Standar protokol terbuka, didapat secara bebas dan dikembangkan dengan tidak bergantung pada *hardware* komputer atau sistem operasi tertentu. Karena ditopang secara luas, TCP/IP sangatlah ideal untuk menggabungkan *hardware* dengan *software* yang berbeda.
- Tidak bergantung pada fisik *hardware* jaringan tertentu. Ini memungkinkan TCP/IP untuk menggabungkan banyak macam jaringan. TCP/IP untuk menggabungkan banyak macam jaringan. TCP/IP dapat dijalankan melalui sebuah Ethernet, token ring, dial-up line, jaringan X.25, dan banyak macam media transmisi fisik.
- Skema pengalamatan yang umum yang memungkinkan peralatan TCP/IP secara unik memberi alamat peralatan lain di seluruh jaringan bahkan jika jaringan tersebut seluas internet dengan jangkuan dunia.
- Protokol-protokol yang berstandar tingkat tinggi untuk pelayanan pemakai yang bersifat konsisten dan tersedia secara luas

TCP/IP terdiri dari empat layer, yaitu:

- *Application Layer*, tempat dimana program-program akan berjalan.

- *Host-To-Host Layer*, digunakan untuk menetukan jenis koneksi yang digunakan antar *host*.
- *Internet Layer*, Untuk menetukan jenis paket apa yang akan digunakan dalam proses pengiriman data
- *Physical Layer*, digunakan untuk mengatur penerimaan atau pengiriman data melewati jalur-jalur fisik.

## 2.2. HTTP Server dan Web Browser

Protokol yang sangat penting dalam internet adalah HTTP, singkatan dari Hyper Text Transfer Protocol. Sedangkan komputer yang menjalankan HTTP ini disebut *Web Server*. Bila suatu sistem operasi mendukung TCP/IP dan multitasking, maka protokol HTTP dapat digunakan. HTTP adalah dasar dari layanan *World Wide Web* (WWW). Transfer dokumen *web* baik berupa text, grafik, audio dan video ataupun gabungannya dari *web server* ke komputer pemakai menggunakan protokol HTTP. Cara kerjanya adalah sebagai berikut: *software browser* (*web browser*) yang di jalankan di komputer pemakai mengajukan permintaan ke *server* HTTP. Kemudian dokumen dengan format HTML (Hyper Text Mark-Up Language) yang sesuai dengan permintaan tersebut akan dikirim ke komputer pemakai *Web browser* lalu akan menampilkan dokumen HTML ke layar pemakai.

Informasi yang disajikan oleh HTTP *server* umumnya bersifat statis, artinya pemakai hanya bisa mengakses ke dokumen dengan format HTML yang terdapat dalam *web server*. Pada umumnya, informasi dalam sebuah organisasi atau perusahaan berupa *database* yang diakses dengan memakai aplikasi khusus. Bila dikehendaki, akses ke dalam *database server* juga dapat dilakukan

Server HTTP (Hyper Text Mark-Up Language) adalah komputer yang bertugas untuk menjalankan protocol HTTP dan menyediakan akses untuk mendistribusikan dokumen, aplikasi *database*. Umumnya, *server* HTTP dikenal dengan nama *web server*. Web *server* bekerja berdasarkan permintaan dari seseorang pemakai yang terhubung ke *web server* tersebut. Permintaan tersebut diolah oleh *web server* dan kemudian hasilnya dikirim kembali ke pemakai yang bersangkutan.

Ada bermacam-macam jenis *software web server* di antaranya: National Center for SuperComputing Applications (NCSA), HTTPd, Apache HTTP Server, Netscape Communications Server, Microsoft Internet Information Server (IIS), Personal Web Server (PWS), dan sebagainya. Dalam penyusunan tugas akhir ini, *software* yang digunakan adalah Microsoft Internet Information Server versi 5.2. Adapun faktor-faktor yang menjadi pertimbangan untuk memilih *software* tersebut adalah:

- Terintegrasi dengan Windows 2000 Server sehingga Internet Information Server (IIS) mudah untuk di-*setup*, mudah pemeliharaannya dan cepat.
- Merupakan *web server* dengan kemampuan yang cukup lengkap di antaranya kemampuan untuk mentransfer dokumen *multimedia*, fasilitas, *log file* dan lain-lain.
- Mendukung bahasa pemrograman JavaScript dan VBScript.
- Memberi kemudahan untuk mendesain suatu *web page* secara interaktif karena pada IIS versi 3.0 ke atas mendukung teknologi Active Server Pages (ASP) yang mempermudah *programmer* untuk mendesain suatu *web page* secara interaktif, tanpa perlu mengkompilasi setiap kali terjadi perubahan pada *web page* tersebut.

*Web Browser* adalah suatu program yang didesain untuk mengambil informasi dari suatu komputer *server* pada jaringan internet. Informasi ini dikemas dalam *page* (dokumen HTML) yang masing-masing memiliki beberapa *link* yang menghubungkan *web* page ke sumber informasi (*page*) yang lain. Jika suatu *link* di-*click*, maka *web browser* akan melihat alamat dari tujuan *link* tersebut dan meminta dokumen HTML dengan alamat yang dimaksud maka dokumen itu akan diberikan kepada *web browser* untuk ditampilkan di layar monitor komputer pemakai. Jika diteruskan, maka *web server* akan memberikan pesan error yang kemudian diteruskan oleh *web browser* ke layar monitor pemakai yang menyatakan bahwa alamat dari tujuan *link* tidak dapat ditemukan.

Pertama kali *web browser* dibuat dengan berbasis teks, yang ditujukan untuk komputer yang menggunakan sistem operasi Unix, contohnya Lynx. Setelah itu muncul *web browser* Mosaic dari NCSA yang berbasiskan grafis dan

mudah digunakan. *Browser* ini dipakai untuk komputer dengan sistem operasi Unix, NeXT, Windows dan Macintosh.

Sekitar tahun 1994, muncullah Netscape versi pertama yang menggantikan kepopuleran Mosaic sebagai *web browser*. Setelah itu salah satu perguruan tinggi terkenal di Amerika Serikat, MIT membangun standard bagi teknologi *web browser* mengarah pada kemampuan untuk mendukung Dynamic Hyper Text Mark-Up Language (DHTML). *Web Browser* yang sangat populer saat ini antara lain Internet Explorer dan Netscape.

## 2.3. HTML

HTML merupakan singkatan dari Hypertext Mark-Up Language. HTML adalah suatu bahasa *mark up* (penandaan) terhadap sebuah dokumen teks yang digunakan untuk membangun sebuah halaman *Web*, yang tidak terikat pada suatu aplikasi tertentu dan dapat dilihat melalui *Web Browser*. HTML mempunyai elemen-elemen tertentu (tag) yang mana tiap-tiap tag tesebut mempunyai fungsi atau perintah tertentu. HTML didefinisikan oleh badan-badan yang yang mengatur Web dan kemudian digunakan oleh *web browser*. Dengan menggunakan HTML, *programmer* dapat menampilkan dan menambah *hyperlink* untuk lompat ke halaman atau dokumen yang lain. teks yang ditulis dalam format hyperlink dapat dipilih oleh pemakai dengan menggunakan *mouse*, begitu dipilih, dokumen yang bersangkutan akan dibawa pada *web browser* pemakai.

### 2.3.1. Struktur Dasar HTML

Sebuah file HTML merupakan file teks murni yang yang mengandung *tag-tag* HTML. Karena merupakan file teks murni, maka file HTML dapat dibuat dengan menggunakan *text editor* sederhana seperti Notepad. Dapat juga dengan menggunakan HTML editor yang bersifat visual seperti FrontPage atau Dreamweaver, yang meskipun dapat digunakan untuk mendesain halaman *web* tanpa harus mengenal *tag* HTML, namun biasanya tetap menyediakan fasilitas untuk menuliskan *tag* HTML secara manual.

Sebuah elemen merupakan bagian yang paling dasar dari HTML. Sebuah elemen terdiri dari *tag* pembuka dan *tag* penutup di mana karakter data diapit oleh

keduanya. Sebuah *tag* diawali oleh tanda lebih kecil (<) dan diakhiri oleh tanda lebih besar (>). *Tag* penutup memiliki tambahan tanda *slash* (/) setelah tanda lebih kecil (<). Beberapa *tag* memerlukan *tag* penutup tapi sebagian lagi tidak memerlukan *tag* penutup.

*Tag* HTML tidak bersifat *case sensitive,* jadi artinya <HTML> sama saja dengan <html>. Penulisan *tag* dengan huruf kapital hanya untuk mempermudah menemukan perbedaan antara teks biasa dengan *tag*.

Secara lengkap, file HTML biasanya mempunyai bagian head dan bagian body, jadi struktur lengkapnya adalah sebagai berikut:

```
<HTML>
<HEAD>
……
</HEAD>
<BODY>
……
</BODY >
</HTML>
```

Elemen dasar dari dokumen adalah *tag* (<HTML>), yang menginformasikan pada *browser* bahwa isi file tersebut ditulis dalam HTML. *Tag* penutup yang berhubungan (</HTML>) adalah *tag* terakhir dari file tersebut. *Tag* (<HEAD>) menandai permulaan dari header dokumen. Header dokumen menggambarkan elemen-elemen yang sesuai untuk semua bagian dari dokumen yang aktif saat itu dan dokumen-dokumen manapun yang berisi atau berhubungan dengan dokumen tersebut. Umumnya elemen TITLE muncul pada header. Internet Explorer menunjukkan teks elemen TITLE pada *bar title*-nya. Sebuah *bar* menu atau gambar yang berulang pada dokumen-dokumen yang lain mungkin muncul pada bagian *header*. Elemen BODY muncul pada awal dari isi utama dokumen. Elemen BODY meliputi *body* teks, gambar-gambar dan obyek-obyek multimedia.

### 2.3.2. Hyperlink

Dengan HTML *hyperlink* antar dokumen dapat dibuat. Sebuah *hyperlink* adalah teks atau gambar apapun yang apabila di-*click* memuat dokumen lain atau bagian lain dari dokumen yang aktif saat itu ke dalam *window Web Browser*. Elemen A atau *anchor* menghubungkan teks atau gambar tersebut ke dokumen

lain atau ke suatu bagian dalam dokumen yang sedang aktif. Sebuah hyperlink tampak sebagai "hot spot" yang dapat di-click (berupa teks atau gambar yang dapat di-click). Untuk membuat *hyperlink*, teks atau gambar dengan *tag-tag anchor* harus dilampirkan dan diikuti dengan atribut HREF=ke alamat yang dituju sebagai berikut:

```
Click <A HREF="//www.petra.ac.id/">di sini</A>
```

### 2.3.3. Form

*Tag* <FORM> merupakan *tag* yang digunakan untuk mendefinisikan sebuah *form*. *Tag* ini mempunyai dua atribut penting yaitu: ACTION dan METHOD.

Atribut ACTION digunakan untuk menentukan URL yang digunakan untuk memproses *form* tersebut. Bila tidak ada URL yang disebutkan, maka URL dari dokumen yang sedang aktif yang digunakan.

Atribut METHOD digunakan untuk menentukan bagaimana cara data dalam *form* tersebut dikirimkan ke *web server*. Ada dua cara bagaimana data dikirimkan ke *web server* yaitu: GET dan POST. Pada METHOD = GET, data dari sebuah *form* akan dikirim melalui alamat URL. Sedangkan pada METHOD = POST, data dari sebuah *form* akan dikirim melalui *header* dari *file* HTML tersebut (tidak terlihat). Contoh:

```
<FORM METHOD=POST ACTION="Login.asp">
.............
</FORM>
```

### 2.3.4. Tag <INPUT>

*Tag* <INPUT> digunakan untuk membuat komponen-komponen yang digunakan untuk meminta informasi dari user seperti: *text box*, *password*, *check box*, *submit button*, *reset button* dan *radio button*. *Tag* <INPUT> mempunyai beberapa atribut penting yaitu: NAME, TYPE, SIZE, MAXLENGTH, VALUE dan CHECKED.

Atribut NAME digunakan untuk menentukan nama dari komponen tersebut untuk membedakan antara satu komponen dengan komponen lainnya.

Atribut TYPE digunakan untuk menentukan tipe dari *input* tersebut. Jika TYPE bernilai teks maka *input* tersebut akan ditampikan dalam bentuk *text box.* Jika TYPE bernilai *password* maka *input* tersebut akan ditampikan dalam bentuk *text box* dengan *mask* untuk menuliskan *password* agar tidak terbaca langsung pada monitor. Jika TYPE bernilai *checkbox* maka *input* tersebut akan ditampilkan dalam bentuk *check box*. Jika TYPE bernilai *radio* maka *input* tersebut akan ditampikan dalam bentuk *radio button.* Jika TYPE bernilai *reset* maka *input* tersebut akan ditampikan dalam bentuk tombol yang berfungsi untuk me*reset* nilai seluruh *input* pada *form* di mana tombol *reset* itu berada. Jika TYPE bernilai *submit* maka *input* tersebut akan ditampikan dalam bentuk tombol yang berfungsi untuk mengirimkan seluruh data yang ada dalam form di mana tombol *submit* tersebut berada.

Atribut SIZE digunakan untuk menentukan ukuran panjang *text box* dalam satuan *pixel.*

Atribut MAXLENGTH digunakan untuk menentukan jumlah maksimum karakter yang dapat dimasukkan dalam *text box*.

Atribut VALUE mempunyai fungsi yang berbeda-beda untuk tiap tipe *input*. Pada *text box,* berfungsi untuk menentukan nilai teks yang tertulis. Pada *check box* atau *radio,* berfungsi untuk menentukan nilai item yang dipilih. Untuk Submit dan Reset, berfungsi untuk menentukan teks yang tertulis pada tombol.

Atribut CHECKED hanya digunakan pada tipe input *check box* dan *radio* untuk menentukan pilihan yang terpilih secara default pada saat halaman HTML dibuka.

### 2.3.5. Tag <SELECT>

*Tag* <SELECT> digunakan untuk membuat *drop down list* yang berisi daftar pilihan yang ditentukan terlebih dahulu. *Tag* <SELECT> mempunyai beberapa atribut penting seperti: NAME dan MULTIPLE.

Atribut NAME berfungsi untuk mendefinisikan nama dari obyek *select* itu sendiri.

Atribut MULTIPLE berfungsi untuk mengizinkan pemilihan pilihan lebih dari satu.

Contoh:

```
<SELECT NAME=”Race”>
        <OPTION SELECTED VALUE=”Human”>Human
        <OPTION VALUE=”Dwarf”> Dwarf
        <OPTION VALUE=”Undead”> Undead
        <OPTION VALUE=”Elf”> Elf
</SELECT>
```

Contoh tersebut menyatakan bahwa *drop down list* tersebut bernama “Race” dan mempunyai empat pilihan yaitu: “Human”, “Dwarf”, “Undead” dan “Elf”. Option “Human” terpilih secara *default* karena terdapat atribut SELECTED di dalamnya.

### 2.3.6. Tag <TEXTAREA>

*Tag* <TEXTAREA> digunakan untuk membuat sebuah *text box* yang mempunyai banyak baris. *Tag* <TEXTAREA> mempunyai beberapa atribut yang penting seperti: NAME, ROWS, COLS, dan VALUE.

Atribut NAME berfungsi untuk mendefinisikan nama dari obyek textarea itu sendiri.

Atribut ROWS berfungsi untuk menentukan jumlah baris textarea tersebut akan ditampilkan.

Atribut COLS berfungsi untuk menentukan lebar textarea tersebut akan ditampilkan.

Atribut VALUE berfungsi untuk menentukan teks awal yang ada dalam textarea tersebut.

### 2.3.7. Tag <FONT>

*Tag* <FONT> digunakan untuk mengatur jenis, ukuran dan warna *font* dari teks yang ditampilkan pada *web browser*. *Tag* <FONT> mempunyai tiga atribut penting yaitu: FACE, COLOR dan SIZE.

Atribut FACE adalah nama dari jenis font yang akan ditampilkan. Nama font ini hendaknya tidak mengandung spasi. Dan apabila mengandung spasi harus diawali dan diakhiri dengan tanda kutip (“).

Atribut COLOR digunakan untuk menentukan warna dari teks yang berada di antara tag <FONT> ini. Format penulisan isi dari atribut ini adalah

sebagai berikut: #RRGGBB. RR adalah dua digit heksadesimal yang mewakili intensitas dari warna merah. GG adalah dua digit heksadesimal yang mewakili intensitas dari warna hijau. BB adalah dua digit heksadesimal yang mewakili intensitas dari warna biru.

Atribut SIZE digunakan untuk menentukan ukuran dari teks yang berada di antara tag <FONT> ini. Contoh:

```
<FONT FACE=Arial COLOR=#CCCC99 SIZE=3>
……..
</FONT>
```

### 2.3.8. Menambahkan Gambar

Halaman HTML akan menjadi lebih menarik dengan adanya tambahan gambar-gambar. Format file gambar yang dapat ditampilkan sepenuhnya tergantung dari jenis *web browser* yang digunakan. Karena itu format file gambar yang digunakan hendaknya adalah format yang umumnya dapat ditampilkan oleh kebanyakan *web browser*. Format file gambar yang umum digunakan adalah JPG dan GIF. Selain karena umum penggunaannya, juga karena ukurannya yang kecil. Format JPG digunakan apabila kombinasi warna dari gambar sangat besar atau melebihi 256 warna atau dimensi dari gambar cukup besar. Format GIF digunakan untuk gambar-gambar yang berdimensi kecil, mempunyai kombinasi warna yang kurang dari 256 warna dan gambar animasi.

Untuk menambahkan gambar ke dalam halaman HTML digunakan *tag* <IMG>. *Tag* <IMG> mempunyai beberapa atribut penting yaitu: SRC, ALIGN, WIDTH, HEIGHT dan ALT.

Atribut SRC digunakan untuk menentukan letak dari file gambar tersebut berada.

Atribut ALIGN digunakan untuk menentukan letak dari teks di sekitar gambar tersebut. Nilainya dapat bervariasi seperti: TOP, BOTTOM, MIDDLE, LEFT dan RIGHT.

Atribut WIDTH digunakan untuk menentukan ukuran lebar dari gambar dalam satuan pixel.

Atribut HEIGHT digunakan untuk menentukan ukuran tinggi dari gambar dalam satuan pixel.

Atribut ALT digunakan untuk menampilkan teks pengganti apabila gambar tidak dapat ditampilkan pada *web browser*. Dan pada *browser* tertentu dapat juga ditampilkan sebagai *tool tip*.

## 2.4. Database dan ODBC

### 2.4.1. Database

*Database* adalah kumpulan dari data komputer yang diintegrasikan, diorganisir dan disimpan dengan cara yang mudah untuk diakses. Untuk itu digunakan *direct access storage*. Integrasi secara logika dari *record* di beberapa *file* disebut konsep *database*. Dalam setiap perancangan dan pembuatan suatu sistem *database* yang baik diperlukan suatu prinsip mendasar yang sangat penting, yaitu: efisiensi dan ketepatan. Suatu *database* dibuat untuk menangani kumpulan data yang bersifat kompleks supaya menjadi lebih ringkas tanpa menghilangkan karakteristik data-data tersebut. Berikut ini beberapa keuntungan yang dapat diperoleh dari penggunaan *database*, antara lain:

- *Redundancy* data diminimumkan, sehingga kemungkinan adanya data yang sama/kembar berkurang.
- Data yang *independence*, sehingga proses *update, insert*, *delete* dapat dilakukan tanpa merubah program yang ada.
- Data lebih terintegrasi, sehingga proses *update, insert*, *delete* menjadi lebih mudah, karena tabel-tabelnya telah terhubung satu dengan yang lain.
- Data dapat diakses dengan lebih cepat, karena data-datanya telah terintegrasi.
- *Security* data menjadi lebih terjamin, karena tersedia beberapa level sekuritas data.

Data model yang saat ini paling sering digunakan dalam *database* adalah *relational data model*. Pengertian akan desain relasi *database* sangatlah penting dalam penggunaan teknologi ini, karena akan membawa suatu perubahan pada cara bagaimana informasi disajikan pada umum. Pada pertengahan tahun 70-an, E.F. Codd yang bekerja pada IBM pada saat itu menulis *A Relational Model of Data for Large Shared Data Banks.* Tulisannya ini membuat revolusi pada industri database dengan menyediakan model untuk organisasi database yang sangat berbeda dengan teori yang lain pada saat itu. Relational Database Model

hanya berhubungan dengan bagaimana data disajikan. Model ini tidak berhubungan dengan bagaimana data disimpan. Data dipresentasikan sebagai tabel-tabel. Tiap kolom disebut *field*. Tiap baris disebut *record*. Kolom-kolom ini dipakai untuk menunjukkan bahwa satu grup informasi yang saling berhubungan diletakkan pada satu tabel. Sebuah tabel adalah kumpulan dari kolom dan baris. Kolom menjelaskan atribut. Tiap baris dari tiap harga dibagi menjadi kolom-kolom yang akan berisi setidaknya satu harga yang unik. Baris-baris ini disebut *record*. Sebuah kolom adalah sebuah koleksi vertikal dari *field-field*, sedangkan baris adalah koleksi horisontal dari *field-field*. Sedangkan sebuah *field* adalah sebuah unit tunggal informasi. *Field* menunjuk pada sebuah kolom dalam suatu baris. Pada *level field*, pemakai berinteraksi dengan tabel dengan memasukkan data. Saat data dimasukkan ke dalam sebuah *field*, data tersebut akan diperiksa kebenaran dan tipe datanya. Sekali tipe data tidak benar maka sebuah pesan kesalahan akan muncul. Ada banyak tipe data yang disediakan oleh program *database* yaitu: *binary*, *date/time*, desimal, *image*, *string*, dan masih banyak yang lain. Informasi ini memberitahu *database* bagaimana menerjemahkan data ke dalam kolom. *Primary keys* adalah kolom yang nilainya secara unik mengidentifikasikan sebuah baris dalam sebuah tabel. Tanpa sebuah *primary key*, desain *relational database* percuma saja. Tiap tabelnya hanya punya satu *primary key*. *Primary key* dapat terdiri dari banyak *field* (*composite key*). *Candidate key* juga merupakan kolom yang mempunyai nilai unik. Dalam satu tabel bisa terdapat banyak *candidate key*. *Candidate key* dapat menjadi *primary key* bila tidak memiliki nilai *null* atau nilai duplikat.

### 2.4.2. Data Flow Diagram (DFD)

DFD merupakan sebuah skema yang menggambarkan jalannya alur-alur data dari suatu proses ke proses selanjutnya. DFD dapat dibagi menjadi beberapa level untuk menggambarkan kompleksitas pergerakan data secara lebih jelas dan teliti. Biasanya suatu DFD dimulai dari *context diagram*, kemudian di-*decompose* menjadi *level* 0, *level* 1, *level* 2,dan seterusnya. Dalam DFD *level* 0, *level* 1, dan 2 dapat dibuat *sub*-DFD yang lebih menggambarkan kelengkapan proses data secara lebih jelas dan mendetail.

DFD memiliki 4 komponen penting, yaitu:

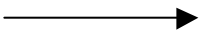
- *Data sources* dan *destinations*

  Simbol dari sumber/*source* atau tujuan/*destination* yang mengirim atau menerima data yang dihasilkan atau digunakan oleh sistem. Biasanya disebut juga *entity,* simbolnya berbentuk persegi dengan terdapat nama *entity* di dalamnya.

Gambar 2.1. Entity

- *Data flow*

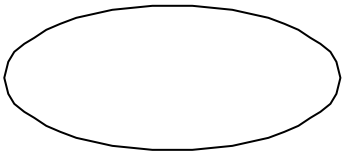
  *Data flow m*enggambarkan aliran data yang terjadi dalam proses, *data store*, dan *entity*. Data yang melalui *data store* maupun *data source and destination* harus melalui *data processing*. Aliran data ini digambarkan dalam bentuk anak panah, panah dari *data flow* diberi label untuk menunjukkan tipe data yang akan mengalami proses tersebut.



Gambar 2.2. Data Flow

- *Process*

  Sebuah *process* menggambarkan transformasi data dari *input* menjadi *output*. Simbolnya dalam bentuk bersegi bersudut bulat, dengan terdapat nama proses di dalamnya.



Gambar 2.3. Process

- *Data Store*

  *Data store* adalah tempat penyimpanan data baik yang bersifat sementara maupun yang tetap.

Gambar 2.4. Data Store

2.4.3. Entity Relationship Diagram (ERD)

*Entity Relationship Diagram* yang kemudian disebut sebagai ERD digunakan untuk menggambarkan skema sistematis mengenai keseluruhan entitas yang terdapat dalam suatu sistem *database*. Disebut *entity relationship diagram* karena E-R diagram ini menggambarkan relasi atau hubungan antar entitas yang ada. E-R diagram biasanya terdiri dari *entity-entity*, relasi yang terdapat di antara *entity* disebut *connectivity.*

*Entity* dapat berupa *environmental element*, *resource*, dan transaksi yang sangat penting, sehingga dimodelkan dalam data. Di dalam E-R diagram, *entity* disimbolkan berupa persegi yang di dalamnya terdapat nama *entity*nya. Relasi adalah hubungan yang terjadi di antara 2 *entity*, biasanya dilambangkan dalam bentuk belah ketupat dengan kata kerja di dalamnya.

Diharapkan dengan melakukan pembentukan ERD, pengkaji dapat lebih memahami bagian-bagian atau entitas yang terdapat dalam suatu perusahaan dan dapat lebih jelas mengenai relasi yang terbentuk anatara entitas-entitas tersebut.

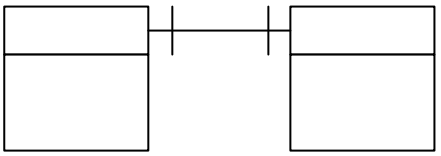
*Entity Relationship Diagram* adalah metode perancangan *database* yang sering dan harus digunakan oleh seorang *programmer* untuk menentukan sistem *database* yang efektif menyelesaikan permasalahan. Dengan menggunakan metode ERD ini, dapat dilihat dengan jelas hubungan antara tabel-tabel dalam database. Dengan melihat ERD dari suatu program dapat dilihat struktur database program tersebut dan selanjutnya dapat dengan mudah untuk meng*upgrade* program tersebut.

Dalam perencanaan ERD dari suatu database, langkah pertama yang harus dilakuakan adalah menyusun elemen-elemennya. Elemen-elemen dari ERD adalah tabel-tabel dan garis-garis penghubungnya. Garis yang menghubungkan antara dua tabel menyatakan relasi kedua tabel tersebut. Dalam struktur *database* terdapat 3 jenis model relasi atau hubungan, antara lain: *One to One Relationship, One to Many Realtionship*, dan *Many to Many Relationship.*

2.4.3.1. One to One Relationship

Relasi yang pertama adalah o*ne to one relationship. One to one relationship* merupakan suatu model hubungan di mana satu anggota *entity* memiliki hubungan dengan satu anggota *entity* pada kelas yang berbeda, di mana terdapat pembagian 2 jenis hubungan, yaitu:

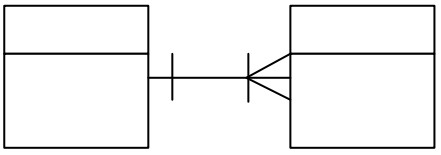
- *Obligator*: suatu keadaan di mana semua anggota dari suatu *entity* harus berpartisipasi atau memiliki hubungan dengan *entity* yang lain.
- *Non-obligator*: suatu keadaan di mana tidak semua anggota dari suatu *entity* harus berpartisipasi atau memiliki hubungan dengan *entity* yang lain.



Gambar 2.5. *One to One Relationship*

2.4.3.2. One to Many Relationship

Relasi kedua adalah o*ne to many relationship. One to many relationship* merupakan suatu relasi atau hubungan antara satu anggota *entity* dengan beberapa anggota *entity* pada kelas yang berbeda. Sama halnya pada *One to One Relationship*, pada relasi yang satu ini juga terbagi ke dalam 2 jenis hubungan, yaitu: *obligator* dan *non-obligator.*

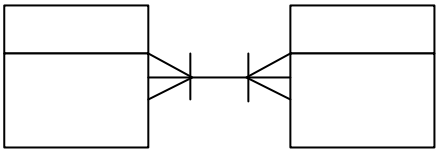


Gambar 2.6. *One to Many Relationship*

2.4.3.3. Many to Many Relationship

Relasi berikutnya adalah m*any to many relationship. Many to many relationship* merupakan hubungan yang terjalin antara beberapa anggota *entity* yang satu dengan beberapa anggota *entity* yang lainnya. Kedua belah pihak dapat

memiliki hubungan lebih dari satu (*many*) dengan beberapa anggota *entity.* Dalam relasi ini juga terdapat 2 jenis hubungan, yaitu: *obligator* dan *non-obligator.*



Gambar 2.7. *Many to Many Relationship*

2.4.4. Teknologi Software Database

Banyak sekali *software* Client/Server dibuat dan dikomersilkan oleh beberapa vendor. Mereka berlomba-lomba untuk mendapatkan penilaian yang tinggi dari masyarakat pengembang aplikasi. Masing-masing vendor menyediakan fasilitas, kemampuan dan kemudahan yang berbeda-beda. Mereka berusaha untuk menciptakan suatu cara yang terbaik, mudah, cepat dan sederhana dalam hal akses *database*, baik dari sisi pemakai maupun *server*.

Teknologi *software database* dikembangkan dalam dua bagian terpisah dan tidak tergantung satu sama lain, yaitu bagian *server* dan bagian pemakai. Masing-masing *software database server* mempunyai format representasi database yang berbeda-beda, sehingga sulit bagi pengembang untuk menyatukan suatu teknologi *database* yang sudah ada dengan suatu teknologi yang baru. Oleh karena itu diperlukan suatu *software* yang diletakkan di antara pemakai dan *server*. Software ini akan berfungsi sebagai penerjemah dari format *database* yang berbeda.

Software *database* untuk platform *server* melakukan beberapa fungsi penting dalam komputasi tersebar, yaitu:

- Memelihara struktur *database*.
- Mengatur manajemen *database* tersebar.
- Administrasi dan manipulasi data (*update, delete insert, select*).
- Eksekusi suatu proses.

Fungsi-fungsi tersebut merupakan fungsi standard yang terdapat pada hampir semua *software database* untuk platform *server*. Selain itu, masing-masing vendor juga menyediakan fasilitas-fasilitas tambahan untuk mendukung

operasional dari *database server*. Berbagai macam *software* database untuk platform *server* yang dibuat oleh beberapa *vendor,* yaitu:

- Oracle Corporation menciptakan produk Oracle Server.
- Microsoft Corporation menciptakan produk Microsoft SQL Server.
- Sybase Corporation menciptakan produk SQL Server Anywhere

Ketiga macam *software* tersebut dapat menerapkan prinsip dari komputasi tersebar. Masing-masing *software* mempunyai kemampuan dan format database yang berbeda-beda sehingga diperlukan suatu perangkat lunak lagi yang disebut dengan *Application Programming Interface.* Software ini diletakkan pada sisi pemakai untuk menerjemahkan suatu format SQL dan atau *database* format yang dapat dimengerti oleh *server* 1(Native API).

Perangkat lunak Middleware dikembangkan agar mendukung implementasi *database* dengan menggunakan teknologi DBMS yang berbeda-beda. Hal ini akan membantu *programmer* dalam mengembangkan program-program aplikasi baru dengan memakai teknologi DBMS yang lain yang berbeda dengan program-program aplikasi yang lama. Dengan adanya perangkat lunak ini, maka programmer tidak perlu lagi melakukan proses migrasi dari format *database* yang lama ke format *database* yang baru.

Sarana ini sering disebut *Application Programming Interface* (API) yang diletakkan pada sisi pemakai. Software ini akan menyediakan format data dan fungsi mapping antara sumber-sumber data yang beraneka ragam (*heterogenous data sources*).

Beberapa contoh perangkat lunak yang dapat digunakan adalah:

- ODBC
- IDAPI
- Oracle Glue

Aplikasi yang ditulis dalam bentuk ODBC API, memanggil fungsi-fungsi ODBC yang ada. *Driver Manager* akan mengaktifkan semua *driver* yang diperlukan, khususnya dalam membuat *data source*. Operasi-operasi yang dilakukan oleh *driver* tersebut adalah:

- Melakukan mapping fungsi-fungsi ODBC ke dalam *Native API.*
- Melakukan ODBC *function call*.

- Melakukan proses submit terhadap SQL ke suatu datasource.
- Mengembalikan hasil ke aplikasi.

### 2.4.5. Application Programming Interface (API)

Untuk mengakses suatu database yang ada di sebuah *server*, dari program aplikasi di komputer pemakai dibutuhkan suatu interface atau sarana antara aplikasi dan *server* tersebut. Dimana sarana tersebut sering disebut sebagai *Application Programming Interface (*API*)* yang diletakkan di sisi pemakai.

Interface yang dipakai dalam tugas akhir ini adalah *Open DataBase Connectivity* (ODBC) karena penggunaannya lebih mudah dan memungkinkan seorang programmer untuk mengakses berbagai jenis *database* yang berasal dari berbagai jenis aplikasi denganan hanya mengganti ODBC *driver*-nya saja.

## 2.5. Active Server Pages (ASP)

*Active Server Pages* (ASP) merupakan suatu *script* yang bersifat *server-side* yang ditambahkan pada HTML untuk membuat sebuah *web* menjadi lebih menarik, dinamis dan interkatif.

ASP adalah *script* yang bersifat *server-side* berarti proses pengerjaan atau pengeksekusian *script* berlansung di *server* bukan di *web browser* client. Dengan kata lain jika ada *web browser* yang mengirimkan permintaan ke *web server*, maka *server* yang menerima permintaan tersebut akan mengeksekusikan setiap *script* yang ada pada file ASP yang diminta tersebut dan hasilnya dikirimkan ke *web browser* tersebut.

ASP adalah suatu ciri dari dan dapat digunakan dengan *web server*-*web server* berikut:

- Microsoft Internet Information Server.
- Microsoft Peer Web Services.
- Microsoft Personal Web Server.

### 2.5.1. Sejarah Singkat Hypertext

Asal mula Web adalah dari linked static content, dan banyak *web*site sekarang ini tetap bersifat statis, yang artinya halaman HTML harus di*edit* secara

manual untuk mengubah apa yang dikirim *web server* ke *browser*. Pada model statis, sebuah *web browser* menggunakan Hypertext Transport Protocol (HTTP) untuk meminta sebuah file HTML dari sebuah *web server*. Server menerima permintaan dan mengirim sebuah halaman HTML ke *web browser*, yang menampilkan halaman tersebut. Walaupun model ini meyediakan akses langsung ke informasi yang terformat rapi, interaksi antara pemakai dan *web server* terbatas. Static pages harus di*edit* manual untuk meng-*update* isinya.

Gateway Interfaces seperti Common Gateway Interface (CGI), Internet Server Application Programming Interface (ISAPI) dan lain sebagainya dapat digunakan untuk menambah isi yang dinamis ke dalam *web*. Dengan interface ini, sebuah *web browser* dapat mengirim sebuah permintaan HTTP untuk aplikasi yang dapat dijalankan daripada sebuah file HTML statis. Server lalu menjalankan aplikasi yang sudah spesifik tersebut. Aplikasi tersebut dapat membaca informasi yang berhubungan dengan permintaan untuk menentukan value mana yang berhubungan dengan permintaan untuk menentukan value mana yang cocok dengan permintaan, seperti value yang dikirimkan pemakai dengan mengisi form HTML. Aplikasi itu kemudian mengolah value menjadi informasi yang berarti dan menghasilkan output di HTML untuk dikirim ke *web browser*. Kerugian program-program gateway adalah program-program tersebut sulit untuk dibuat maupun diubah. Program gateway tidak terintegrasi ke dalam file-file HTML.

ASP dapat digunakan untuk memasukkan *script -script* yang dapat dijalankan langsung ke dalam file-file HTML. Pengembangan HTML dan *script* ing menjadi proses yang sama, yang memungkinkan untuk memfokuskan pada tampilan *web*site dengan elemen-elemen yang dinamis. Aplikasi-aplikasi ASP adalah:

- Terintegrasi seluruhnya dengan file-file HTML.
- Mudah dibuat, tanpa harus mengkompilasi atau melakukan linking program secara manual.
- Berorientasi pada obyek dan extensible dengan komponen-komponen ActiveX *server*.

Hal ini menguntungkan karena memungkinkan *web* provider untuk menyediakan aplikasi-aplikasi bisnis yang interaktif daripada hanya memamerkan

isinya. Misalnya, sebuah biro perjalanan dapat berbuat lebih jauh daripada hanya menunjukkan jadwal penerbangan karena ASP memungkinkan pelanggan mereka untuk mengetahui penerbangan yang ada, membandingkan harga dan bahkan melakukan reservasi. Aplikasi ASP menjadi mudah dengan menggunakan ASP *scripting* untuk mengembangkannya.

### 2.5.2. Struktur Script ASP

Sebuah file ASP merupakan sebuah file text murni yang di dalamnya dapat berisi kombinasi manapun dari:

- Text
- Tag-tag HTML
- Script ASP.

Jadi dapat dikatakan bahwa file ASP sebenarnya adalah file HTML biasa yang ke dalamnya ditambahkan *script* ASP. Jika file HTML biasanya mempunyai ekstensi .htm atau .html, maka jika diberi tambahan *script* ASP di dalamnya ekstensi tersebut tinggal diubah menjadi .asp.

Script ASP dapat diletakkan dimana saja dalam sebuah file, bahkan diluar *tag* <HTML>. Untuk membedakan atau memisahkan *script* ASP dengan teks dan *tag-tag* HTML maka digunakan suatu tanda yang disebut *delimiter*. Delimiter adalah suatu karakter atau kumpulan karakter yang mengawali dan mengakhiri suatu tag atau *script* . Untuk HTML tag *delimiter*nya adalah karakter lebih kecil (<) dan karakter lebih besar (>). Untuk *script* ASP *delimiter* yang digunakan adalah gabungan dari 2 karakter yaitu <% untuk mengawali *script* ASP dan %> untuk mengakhiri *script* ASP. Contoh:

```
<HTML>
<BODY>
This page was last refreshed at <%=Now%>.
</BODY>
</HTML>
```

Pernyataan Now berfungsi untuk menampilkan waktu *server* pada saat fungsi Now tersebut dipanggil. Jadi jika file ASP tersebut dijalankan maka di *web browser* akan terlihat kurang lebih sama seperti ini:

```
This page was last refreshed at 1/1/2002 12:00:00 AM.
```

Script ASP bahkan dapat juga dikombinasikan dengan HTML tag. Contoh:

```
<HTML>
<BODY>
<% FOR I = 1 TO 5 %>
<FONT SIZE = <%=I%> >
SELAMAT DATANG !<BR>
</FONT>
<% NEXT %>
</BODY>
</HTML>
```

Dalam hal ini, *web server* mengembalikan value dari fungsi VBScript "Now" ke *web browser* bersama dengan text.

Sebuah statement dalam VBScript dan bahasa *script* ing lain adalah unit yang lengkap secara syntax yang mengekspresikan sebuah aksi, deklarasi atau definisi. Contoh statement yang umum pada VBScript:

```
<%
If  Time >= #12:00:00 AM# AndTime < 12:00:00 PM# Then
   Greeting="Good Morning!"
Else
   Greeting="Hello!"
End If
%>
```

Statement ini menyimpan baik value "Good Morning!" maupun "Hello!" ke dalam variabel Greeting. Statement tidak mengirim value manapun ke *browser* pemakai.

Text HTML dapat disisipkan di antara bagian-bagian statement seperti pada contoh berikut:

```
<font color="Green">
<%If  Time >= #12:00:00 AM# And Time < 12:00:00 PM# Then%>
Good Morning!
<%Else%>
Hello!
<%End If%>
</font>
```

Bila ada permintaan yang untuk *script* tersebut pada waktu antara tengah malam sampai sebelum tengah hari, maka *web server* akan mengirim HTML yang

mengikuti kondisi “Good Morning!” ke *web browser*, bila tidak ia mengirimkan HTML yang termasuk dalam kondisi “Hello!” ke *web browser*.

Statement, perintah, expression dan prosedur yang digunakan dalam delimiter *script* harus valid untuk *scripting* language yang aktif. ASP dapat juga menjalankan *scripting* language selain VBScript dengan menggunakan tag-tag *script* HTML <SCRIPT> dan </SCRIPT> bersama dengan atribut LANGUAGE dan RUNAT untuk mengerjakan *script* dalam bahasa yang telah ditentukan tersebut. Contoh file ASP berikut menggunakan JavaScript:

```
<HTML>
<BODY>
<%Call MyFunction%>
</BODY>
</HTML>
<SCRIPT RUNAT=Server LANGUAGE=JScript>
 function MyFunction()
{
        Response.Write(“MyFunction Called”)
}
</SCRIPT>
```

Yang perlu diingat adalah bahwa output expression atau perintah lainnya yang bukan merupakan bagian dari *tag* <SCRIPT> tidak boleh dimasukkan dalam *tag* <SCRIPT>.

### 2.5.3. Bahasa Script ASP

Script ASP dapat ditulis dalam dua bahasa yaitu: Microsoft VBScript dan Microsoft JScript. Bahasa yang diasumsikan oleh ASP *web server* apabila tidak ada deklarasi bahasa yang dipilih adalah Microsoft VBScript.

Dalam satu file ASP digunakan satu bahasa primer yang berlaku untuk seluruh isi file tersebut.

Cara pendeklarasian bahasa primer dalam sebuah file ASP adalah sebagai berikut:

```
<%@LANGUAGE=VBScript%>
```

atau

```
<%@LANGUAGE=JScript%>
```