BAB II

LANDASAN TEORI

2.1. *Matrix*

2.1.1. Sejarah *Matrix*

Pembelajaran *Matrix* telah sejak lama ada. *Latin Squares* dan *Magic Squares* telah dikenal sejak jaman prasejarah. *Matrix* mempunyai sejarah panjang tentang aplikasi dalam memecahkan persamaan linear. Beberapa huruf china penting antara *300BC* sampai *AD200*, sembilan bab dalam seni matetamika (*Chiu Chang Suan Shu*) adalah contoh pertama penggunaan metode *matrix* dalam memecahkan persamaan simultan. Pada bab ketujuh "*Too Much and Not Enough*", konsep penentu pertama terlihat hampir 2000 tahun sebelum penemuan oleh Ahli matematik Jepang *Seki Kowa* pada tahun 1683 atau *Leibniz Gottfried* Jerman (seseorang yang telah dihargai dengan penemuan tentang kalkulus/diferensial, yang terpisah namun serempak dengan *Isaac Newton*). *Leibniz* mengembangkan teori determinan pada tahun 1693. *Cramer* mengembang;kan teori tersebut lebih lanjut, mempresentasikan aturan *Cramer's* pada tahun 1750. *Carl Friedrich Gauss* dan *Wilhelm Jordan* mengembangkan *Gauss-Jordan* penghapusan pada tahun 1800. Istilah *matrix* yang pertama tahun 1848 dengan *J. J. Sylvester. Cayley, Hamilton, Grassmann, Frobenius* dan *Von Neumann* adalah antara para ahli matematik yang sudah terkenal.

2.1.2. Pengenalan *Matrix*

Matrix adalah suatu kumpulan besaran (variabel dan konstanta) yang dapat dirujuk melalui *indek*-nya, yang menyatakan posisinya dalam representasi umum

yang digunakan, yaitu sebuah tabel persegi panjang. *Matrix* merupakan suatu cara visualisasi variabel yang merupakan kumpulan dari angka-angka atau variabel lain, misalnya *vektor*. Dengan representasi *matrix*, perhitungan dapat dilakukan dengan lebih terstruktur. Pemanfaatannya misalnya dalam menjelaskan persamaan linier, transformasi koordinat, dan lainnya. *Matrix* seperti halnya variabel biasa dapat dimanipulasi, seperti dikalikan, dijumlah, dikurangkan dan didekomposisikan.

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \quad \text{... (P. 2.1)}$$

1. Penjumlahan dan pengurangan *matrix*.

Penjumlahan dan pengurangan *matrix* dapat dilakukan dengan mengoperasikan komponen *matrix* pada letak yang sama, atau dilambangkan dengan : $a_{ij} \pm b_{ij} = c_{ij}$

atau dalam representasi dekoratifnya :

$$\begin{bmatrix} 1 & 3 & 2 \\ 1 & 0 & 0 \\ 1 & 2 & 2 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 5 \\ 7 & 5 & 0 \\ 2 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1+0 & 3+0 & 2+5 \\ 1+7 & 0+5 & 0+0 \\ 1+2 & 2+1 & 2+1 \end{bmatrix} = \begin{bmatrix} 1 & 3 & 7 \\ 8 & 5 & 0 \\ 3 & 3 & 3 \end{bmatrix} \quad \text{... (P. 2.2)}$$

2. Perkalian Skalar.

Perkalian *matrix* dilakukan dengan mengalikan setiap elemen dari *matrix*, sebagai contoh :

$$2\begin{bmatrix} 1 & 8 & -3 \\ 4 & -2 & 5 \end{bmatrix} = \begin{bmatrix} 2\times 1 & 2\times 8 & 2\times -3 \\ 2\times 4 & 2\times -2 & 2\times 5 \end{bmatrix} = \begin{bmatrix} 2 & 16 & -6 \\ 8 & -4 & 10 \end{bmatrix} \quad \text{... (P. 2.3)}$$

Perkalian 2 buah *matrix* digambarkan jika hanya jika jumlah kolom *matrix* kiri sama dengan jumlah baris *matrix* kanan, sebagai contoh :

$$\begin{bmatrix} 1 & 0 & 2 \\ -1 & 3 & 1 \end{bmatrix} \times \begin{bmatrix} 3 & 1 \\ 2 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} (1\times 3+0\times 2+2\times 1) & (1\times 1+0\times 1+2\times 0) \\ (-1\times 3+3\times 2+1\times 1) & (-1\times 1+3\times 1+1\times 0) \end{bmatrix} = \begin{bmatrix} 5 & 1 \\ 4 & 2 \end{bmatrix} \quad \text{... (P. 2.4)}$$

2.1.3. Teori *Matrix* Grafik

1. *Adjacency Matrix* adalah sebuah *matrix* bujur sangkar yang mewakili suatu grafik, dengan a_{ij} bukan nol jika *vertex* i dan *vertex* j adalah bersebelahan.

2. *Biadjacency Matrix* adalah sebuah kelas khusus mengenai *matrix adjacency* yang menguraikan grafik dua partai.

3. *Degree Matrix* adalah sebuah diagonal matrix yang menggambarkan tingkat derajat *vertex* dalam suatu grafik.

4. *Incidence Matrix* adalah sebuah matrix menggambarkan sebuah hubungan antara 2 kelas objek (biasanya puncak dan ujung dari konteks grafik).

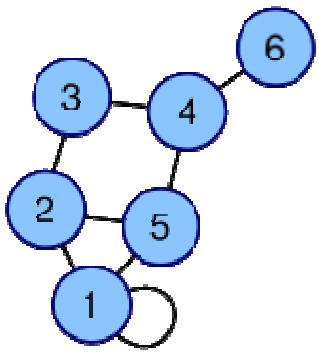
5. *Laplacian Matrix* adalah sebuah *matrix* yang mirip dengan *degree matrix* kurang adjacency matrix untuk sebuah grafik yang digunakan untuk menemukan banyaknya pohon dalam grafik.

6. *Seidel Adjacency Matrix* adalah sebuah *matrix* serupa dengan *adjacency matrix* tetapi dengan -1 *adjacency,* +1 untuk bukan *adjacency,* 0 dalam diagonal.

2.1.3.a. *Adjacency Matrix.*

Pada ilmu matematika dan ilmu komputer, *adjacency matrix* pada grafik G terarah atau tidak terarah, dalam *n* dimana *n* x *n matrix non diagonal* a_{ij} adalah nomor dari *vertex* i ke j, dan *diagonal* a_{ij} adalah dua kali perulangan dalam *vertex* i atau hanya sekali perulangan (pemakaian berbeda, tergantung dalam keperluan matematikal, artikel ini mengarahkan konvensi terdahulu untuk

grafik tidak terbatas. Meskipun grafik terarah selalu mengikuti yang terakhir). Terdapat sebuah *adjacency matrix* unik untuk grafik dan *matrix* ini bukan *adjacency matrix* dalam beberapa grafik. Dalam kasus tertentu mengenai grafik sederhana terbatas, *adjacency matrix* adalah sebuah (0,1) - *matrix* dengan nol dalam diagonal

Grafik	Adjacency Matrix
	1 1 0 0 1 0 1 0 1 0 1 0 0 1 0 1 0 0 0 0 1 0 1 1 1 1 0 1 0 0 0 0 0 1 0 0

Gambar 2.1 ***Adjacency Matrix***

2.1.3.b. ***Biadjacenty Matrix.***

Pada ilmu matematika dan ilmu komputer, *biadjacency matrix* pada grafik G dengan **n** *vertex* hitam dan m *vertex* putih adalah *matrix* **n x m** dimana a_{ij} adalah jumlah *edges* digabung *vertex* i putih dengan *vertex* j hitam. Pada kasus tertentu *biadjacency matrix* adalah (0,1)-*matrix*. *Adjacency Matrix* ***A*** pada sebuah grafik *bipartite* dengan *biadjacency matrix* B, di dapat dari :

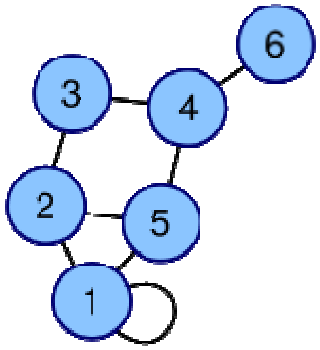
$$A = \begin{pmatrix} 0 & B \\ B^T & 0 \end{pmatrix}. \qquad \text{... (P. 2.5)}$$

Hubungan antara grafik *bipartite* dan *biadjacency matrix* dipelajari pada teori *grafik spectral*.

2.1.3.c. *Degree Matrix.*

Pada ilmu matematika sub teori grafik, *Degree Matrix* adalah diagonal *matrix* yang mengandung informasi tentang derajat *vertex*.Berasal dari *G=(V,E)* dengan $\|V\| = n$, *Degree Matrix* D untuk G adalah ***n x n*** *square matrix* sebagai : $d_{i,j} := \begin{cases} \deg(v_i) & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$

Contoh :

Grafik *Vertex*	*Degree Matrix*
	$\begin{pmatrix} 4 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$

Gambar 2.2 *Degree Matrix*

2.1.3.d. *Incidence Matrix.*

Pada ilmu matematika, *incidence matrix* adalah *matrix* yang menunjukkan hubungan antar dua kelas objek. Jika kelas utama X dan yang kedua Y, *matrix* mempunyai satu baris untuk elemen X dan satu kolom untuk elemen Y. Masukan dalam baris x dan kolom y adalah 1 jika x dan y terhubung dan 0 jika bukan.

2.1.3.e. *Laplacian Matrix.*

Pada ilmu mathematika bidang teori grafik *Matrix Laplacian*, kadang-kadang disebut *admittance matrix* atau *Kirchhoff Matrix*, adalah suatu penyajian *matrix* suatu grafik. Bersama-sama dengan teori *Kirchhoff* dapat digunakan untuk meng-kalkulasi banyaknya memutar pohon untuk grafik yang ditentukan.

Laplacian suatu grafik ***G*** digambarkan sebagai ***L*: = *D* − *A*** dengan D *degree matrix* ***G*** dan A *adjacency* matrix ***G***. Lebih jelasnya diketahui grafik ***G*** dengan ***n*** *vertex*, matrix $L := (l_{i,j})_{n \times n}$

$$l_{i,j} := \begin{cases} \deg(v_i) & \text{if } i = j \\ -1 & \text{if } i \neq j \text{ and } v_i \text{ adjacent } v_j \\ 0 & \text{otherwise} \end{cases} \quad \text{... (P. 2.6)}$$

2.1.3.f. *Seidel Adjacenty Matrix*.

Pada ilmu matematika dan Teori *Matrix* Grafik, *Seidel Adjacency Matrix* pada grafik sederhana ***G*** (disebut juga *Seidel Matrix*) adalah *matrix* simetris dengan baris dan kolom untuk masing-masing *vertex*, mempunyai 0 dalam diagonal dan posisi yang bersesuaian dengan puncak V_i dan V_j, -1 jika *vertices* bersebelahan dan +1 jika tidak bersebelahan. *Seidel Matrix* dikenalkan oleh *Van Lind* dan *Seidel* pada tahun 1966 dan secara aktif dimanfaatkan oleh *Seidel* dan *Coauthors*.

2.2. WEB

Word Wide Web (WWW) lebih dikenal dengan *web*, merupakan salah satu layanan yang didapat oleh pemakai komputer yang terhubung ke internet.

Web pada awalnya adalah ruang informasi dalam internet, dengan menggunakan teknologi *hypertext*, pemakai dituntun untuk menemukan informasi dengan mengikuti *link* yang disediakan dalam dokumen *web* yang ditampilan dalam *browser web.*

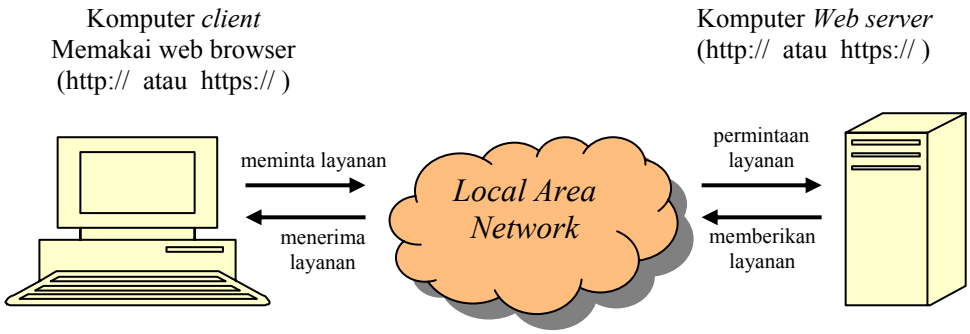
Sejalan dengan perkembangan teknologi informasi, aplikasi *web* selain digunakan untuk mendapatkan informasi juga digunakan sebagai sarana untuk mempermudah pekerjaan seorang *administrator* web dalam mengelola data atau

pun sarana pertukaran file seperti photo maupun *video*. Berikut adalah beberapa alasan kenapa aplikasi berbasis *web* digunakan:

1. Akses informasi mudah.
2. Mempermudah dalam mengkonfigurasi data rute.
3. Informasi mudah didistribusikan.
4. Bebas platform, dalam hal ini informasi atau pengeksekusian aplikasi dapat dikerjakan atau disajikan oleh *web browser* pada sistem operasi mana saja karena adanya standar dokumen berbagai tipe data dapat disajikan.

Prinsip dasar cara kerja dari aplikasi berbasis *web* adalah sebagai berikut :

1. Aplikasi *web* disimpan dalam dokumen yang disebut dengan halaman-halaman *web (web pages).*
2. *Web page* kemudian disimpan dalam komputer yang disebut dengan *web server,* komputer *web server* ini akan melayani permintaan dari *client* yang akan menggunakan *aplikasi web.*
3. Komputer *client* melakukan eksekusi aplikasi melalui *web browser* yang terdapat pada komputer *client* dengan memasukkan alamat dari aplikasi *web* yang ditempatkan di *web server.*



Gambar 2.3 Cara kerja aplikasi berbasis *web*

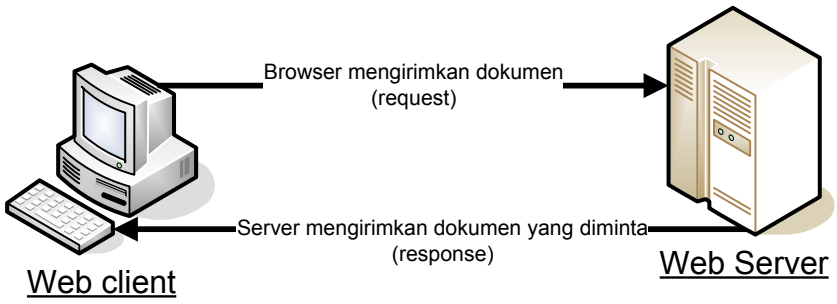
2.3. PHP

2.3.1. Sejarah *PHP*

PHP pertama kali dibuat oleh Rasmus Lerdorf pada tahun 1995. Pada saat tersebut *PHP* adalah sekumpulan script yang digunakan untuk mengolah data *form* (hasil input user) dari *web*. Perkembangan selanjutnya adalah Rasmus melepaskan kode sumber tersebut dan menamakannya *PHP*/*FI*, pada saat tersebut kepanjangan dari *PHP*/*FI* adalah *Personal Home Page/Form Interpreter*. Dengan pelepasan kode sumber ini menjadi *open source*, maka banyak programmer yang tertarik untuk ikut mengembangkan *PHP*. *PHP* 3.0 ditulis ulang dari awal. Dia telah memiliki peningkatan yang sepantasnya dengan kode yang lebih kuat dan konsisten dibandingkan dengan versi 2.0. 3.0 juga lebih cepat secara signifikan, dan menggunakan sedikit memori. *PHP* 4 dengan *Zend engine* yang terintegrasi telah meningkatkan performansi dan kapabilitas dari *PHP* secara besar, tetapi perawatan sangat besar telah dilakukan untuk menghilangkan kerusakan pada kode yang ada sekecil mungkin. *PHP* 5 *Zend Engine* 2 yang terintegrasi telah secara besar meningkatkan daya guna dan kemampuan *PHP*, tetapi pemeliharaan yang besar telah dilakukan untuk kerusakan sekecil mungkin pengkodean sebisa mungkin. Jadi bermigrasi kode Anda dari *PHP* 4 ke *PHP* 5 akan sangat mudah.

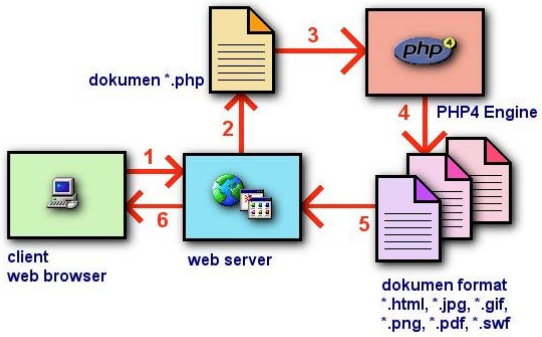
2.3.2. Pengenalan PHP

Jika dilihat dari kerjanya, pemrograman berbasis internet (*web*) dapat digolongkan menjadi dua bagian, yaitu pemrograman *client-side* dan pemrograman *server-side*.



Gambar 2. 4. Konsep kerja Web Client dan Web Server

Pemrograman *server-side* merupakan pemrograman yang diolah pada sisi server (*server-side*). Berbeda dengan model pemrograman *client-side* akan diolah atau diterjemahkan oleh komputer client. *PHP* merupakan salah satu bahasa pemrograman *server-side* diantara beberapa pemrograman yang ada. Sejak diluncurkan, *PHP* mendapat respon yang sangat baik dari kalangan pengembang aplikasi *web*. Kemudahan untuk dipahami, serta sintaknya yang mirip bahasa *C* menjadikan pemrograman ini cepat dikenal oleh kalangan luas.



Gambar 2. 5. Ilustrasi proses PHP

2.3.3. *Array* pada PHP

Dalam banyak aplikasi *Web*, seringkali membutuhkan sebuah variabel yang mampu menyimpan beberapa nilai sekaligus secara terorganisir, sehingga suatu saat dapat memperoleh kembali nilai-nilai tersebut dengan mudah dan

dalam bentuk yang teratur dan terorganisir. Hal ini sangat penting karena dalam banyak kasus membutuhkan penyimpan nilai sementara yang terorganisir dengan baik, dan *database* sangat mahal untuk melakukan hal tersebut. Seperti untuk koneksi, *query* dan seterusnya yang akan memperlambat proses apabila menggunakan *database*. Di sinilah *array* dibutuhkan, yaitu sebuah tipe variabel yang mampu menyimpan beberapa nilai sekaligus bahkan dalam tipe berbeda yang terorganisir dengan baik. Sehingga dalam bahasa pemrograman manapun termasuk *PHP* senantiasa ditemukan *array* sebagai salah satu tipe variabelnya. *Array* di *PHP* adalah tipe variabel spesial, karena dengan *array* dapat diakses sekumpulan nilai dengan menggunakan satu nama variabel. Mekanisme pengaksesan ini diatur dalam bentuk pemetaan (*mapping*) dari sekumpulan nilai (*values*) ke sekumpulan kunci (*keys*) atau indeks. Sebagai contoh, variabel array $nama dapat memetakan 1, 2 dan 3 masing-masing ke “Eric S Raymond”, “Richard Stallman” dan “Bruce Perens”, sehingga dapat dinyatakan sebagai: **$nama[1] = "Eric S Raymond"; $nama[2] = "Richard Stallman"; $nama[3] = "Bruce Perens";** Dimana ekspresi 1, 2 dan 3 adalah indeks, sementara “Eric S Raymond”, “Richard Stallman” dan “Bruce Perens” adalah nilai-nilainya. Dengan *array*, dapat dilakukan hal di atas dengan lebih baik efektif dan efisien. Untuk 900 data nama, cukup melakukan iterasi seperti berikut:

```
for ($k=0;$k<900;$k++){   print "Nama ke $k: $nama[$k] <br>";  }
```

Singkat, sederhana, tetapi ampuh. Selain itu, banyaknya fungsi *PHP* untuk memanipulasi *array* membuat tipe variabel ini sangat membantu kerja dalam membuat aplikasi *web*.

2.4. MySQL

Database MySQL merupakan sistem basis data yang sangat terkenal dan bersifat *Open Source* artinya bahwa semua orang diizinkan menggunakan dan memodifikasi software tersebut. *MySQL* dibangun, didistribusikan, dan didukung oleh *MySQL AB*. Sebenarnya software *MySQL* mempunyai dua macam lisensi, yaitu yang pertama bersifat *Open Source* dengan menggunakan *GNU* (*General Public Licence*) dan lisensi kedua itu berupa lisensi komersial standar (*standard commercial licence*) yang dapat dibeli dari *MySQL AB*. *MySQL* merupakan sistem manajemen *database* atau basis data terhubung (*relational database management system*). *Database* terhubung menyimpan data pada tabel – tabel terpisah. Hal tersebut akan menambah kecepatan dan fleksibilitasnya. Kata *SQL* pada *MySQL* merupakan singkatan dari "*Structured Query Language*". *SQL* merupakan bahasa standar yang digunakan untuk mengakses database dan ditetapkan oleh *ANSI* / *ISO SQL Standard*.

Fitur yang ada pada *MySQL* adalah sebagai berikut :

1. Ditulis dalam bahasa C dan C++,
2. Bekerja dalam berbagai platform (misalnya *Mac OS X*, *Solaris*, *Sun OS*, *UNIX*, *Novel Netware*, *Windows*, dan lain-lain),
3. Menyediakan mesin penyimpan (*engine storage*) transaksi dan nontransaksi,
4. *Server* tersedia sebagai platform program yang terpisah untuk digunakan pada lingkungan jaringan klien/server,
5. *MySQL* mempunyai *library* yang dapat ditempal pada aplikasi yang berdiri sendiri (*standalone application*) sehingga aplikasi tersebut dapat digunakan pada komputer yang tidak mempunyai jaringan,

6. Mempunyai sistem *password* yang fleksibel dan aman,
7. Dapat menangani basis data dalam skala besar. Besar data dalam server *MySQL* dapat berisi 50 juta *record*,
8. Klien dapat terkoneksi ke *MySQL Server* menggunakan *socket TCP/IP* pada platform manapun,
9. *Software* dapat mengirim pesan kesalahan ke klien dalam berbagai bahasa.

MySQL mempunyai logo berupa ikan lumba – lumba yang bernama "*Sakila*". Nama tersebut merupakan kontribusi dan pemenang kontes "nama lumba – lumba" yang diadakan oleh *MySQL AB*. "*Sakila*" berasal dari *SISwati*, bahasa lokal Swaziland, "*Sakila*" juga merupakan nama kota di Arusha Tanzania.