**BAB II**

**LANDASAN TEORI**

**2.1 Tinjauan pustaka**

 Untuk mendukung teori-teori yang digunakan dalam penyusunan skripsi ini, penulis menggunakan beberapa sumber terkait yang akan menjelaskan tentang teori yang di maksud. Adapun teori tersebut antara lain :

1. **Konsep dasar sistem informasi**

 Dalam menyusun dan merancang sistem informasi, terdapat beberapa konsep yang dapat dijadikan sebagai acuan dan landasan, dimana konsep-konsep tersebut merupakan teori yang berhubungan dengan perancangan sistem informasi serta pemecahan masalah yang ada.

1. **Pengertian sistem informasi**

Menurut O’Brian dalam Yakub (2005:17) “Sistem informasi (*information system)* merupakan kombinasi teratur dari orang-orang, perangkat keras (*hardware*), perangkat lunak (*software),* jaringan komunikasi dan sumber daya data yang mengumpulkan, mengubah dan menyebarkan informasi dalam sebuah organisasi”.

Menurut Leitch dan Davis dalam Jogiyanto (2005:11) “Sistem informasi adalah suatu sistem di dalam organisasi yang mempertemukan kebutuhan pengolahan transaksi harian, mendukung operasi, bersifat manajeril dan kegiatan strategi dari suatu organisasi dan menyediakan pihak luar tertentu dengan laporan-laporan yang di perlukan”.

1. **Pengertian sistem**

Menurut McLeod dalam Yakub (2004:1) “Sistem adalah sekelompok elemen-elemen yang terintegrasi dengan tujuan yang sama untuk mencapai tujuan”

Gerald dkk dalam Jogiyanto (2005:1) mendefinisikan “Suatu sistem adalah suatu jaringan kerja dari prosedur-prosedur yang saling berhubungan, berkumpul bersama-sama untuk melakukan suatu kegiatan atau untuk menyelesaikan suatu sasaran tertentu penekanan sistem terhadap prosedur lebih lanjut. Stallings dalam Jogiyanto (2005:2) mengartikan prosedur sebagai berikut: “Suatu prosedur adalah urutan-urutan yang tepat yang menerangkan apa (*what)* yang harus dikerjakan, siapa (*who)* yang mengerjakan kapan (*when*) dikerjakan dan bagaimana (*how)* mengerjakannya”.

Pendapat lain mengenai pengertian sistem yang dikemukakan oleh Jogiyanto (2005:2) “Sistem merupakan kumpulan dari elemen-elemen yang berinteraksi untuk mencapai suatu tujuan tertentu”. Pengertian sistem ini lebih menekankan pada komponen. Pendekatan sistem yang menekankan pada elemen akan lebih memudahkan dalam mempelajari sistem untuk tujuan analisis dan perancangan suatu sistem menurut Jogiyanto (2005:23).

1. **Pengertian informasi**

menurut Mcleod dalam Yakub (2004:8) “informasi *(information)* adalah data yang di olah menjadi bentuk lebih berguna dan lebih berarti bagi yang menerimanya. Informasi jugadisebut data yang di proses atau data yang memiliki arti”.

1. Akurat

 Akurat berarti informasi harus bebas dari kesalahan dan tidak bias atau menyesatkan akurat juga berartiinformasi harus jelas mencerminkan maksudnya. Informasi harus akurat karena dari sumber informasi sampai ke penerima informasi kemungkinan banyak terjadi gangguan *(noise*) yang dapat merubah atau merusak informasi tersebut.

1. Tepat pada waktunya

Tepat pada waktunya berarti informasi yang datang pada penerima tidak boleh terlambat, informasi yang sudah usang tidak akan mempunyai nilai lagi. Karena informasi merupakan landasan di dalam mengambil keputusan. Bila pengambilan keputusan terlambat, maka dapat berakibat fatal untuk organisasi.

1. *Relevan*

Relevan berarti informasi tersebut mempunyai manfaat untuk pemakai nya.

Menurut Presman (2010:39) “*watterfall* adalah model klasik yang bersifat sistemati, berurutan dalam pengembangan software”.

Tahapan–tahapan dalam *the waterfall* model secara ringkas adalah sebagai berikut :

1. Tahap investigasi, dilakukan untuk menentukan apakah terjadi suatu masalah atau adakah peluang suatu sistem informasi dikembangkan. Pada tahapan ini studi kelayakan perlu di lakukan untuk menentukan apakah sistem informasi yang akan dikembangkan merupakan solusi yang layak.
2. Tahap analisis, bertujuan untuk mencari kebutuhan pengguna dan organisasi serta menganalisa kondisi yang ada (sebelum diterapkan sistem informasi yang baru).
3. Tahap desain, bertujuan menentukan spesifikasi detail dari komponen-komponen sistem informasi dan produk-produk informasi yang sesuai dengan hasil tahap analisis.
4. Tahap implementasi, merupakan tahapan untuk mendapatkan atau mengembangkan *hardware* dan *software* (pengkodean program), melakukan pengujian, pelatihan dan perpindahan ke sistem baru.
5. Tahap perawatan (*maintenance*), dilakukan ketika sistem informasi sudah di operasikan. Pada tahap ini, dilakukan monitoring proses, evaluasi dan perubahan (perbaikan) bila diperlukan.

INVESTIGASI

ANALISIS

DESAIN

IMPLEMENTASI

PERAWATAN

Sumber : Presman (2010:39)

**Gambar II.1.**

**The Waterfall Model**

1. ***E-commerce***

Menurut wong (2010:1) “pengertian dari *electronic* *commerce* adalah pembelian, penjualan dan pemasaran barang serta jasa melalui sistem elektronik”.

1. Internet

Menurut Darma dkk (2009:1) “*internet* adalah rangkaian komputer yang terhubung satu sama lain”. Internet merupakan sistem komputer umum, yang berhubung secara global dan menggunakan TCP/IP sebagai protokol pertukaran paket (*packet switching communication protocol).*

TCP (*Transmision Control Protocol)* bertugas untuk memastikan bahwa semua hubungan berkerja dengan benar, sedangkan IP (*interner protocol)* yang mentransmisikan data dari satu komputer ke komputer lain. TCP/IP secara umum berfungsi memilih rute terbaik transmisi data, memilih rute alternatif jika suatu rute tidak dapat digunakan, mengatur dan mengirimkan paket-paket pengiriman data

1. MySQL (*My Structured Query Language)*

Menurut Sukarno (2006b:3) “MySQL merupakan perangkat lunak untuk sistem manajemen *database (Database Management System)”.* Karena sifatnya yang open source dan memilki kemampuan menampung kapasitas yang sangat besar, maka MySQL menjadi *database* yang sangat popular di kalangan *programer web.* MySQL dapat dijalankan dalam 2 *operating system* yang sangat popular saat ini, yaitu: Windows dan Linux.

1. HTML *(Hyper Text Markup Language)*

Menurut Supardi (2010:37) HTML akronim dari *Hyper Text Markup Language.* “HTML merupakan suatu *script* atau program yang bertujuan untuk program *website* atau *homepage”.* Mengapa disebut *hypertext,* karena pemakai dapat membaca tampilan *website* tanpa harus baris demi baris dapat meloncat dari satu baris ke baris yang lain. Disebut *markup laguage* karena susunan instruksinya banyak menggunakan tanda-tanda yang disebut *tag.*

Banyak alasan kita harus mempelajari HTML, antara lain :

1. HTML sebagai bahasa *standart* dalam pembuatan *homepage website.* Seperti halnya anda ingin belajar Borland Delphi, anda lebih baik belajar dahulu bahasa pascal, jika anda ingin belajar Visual Foxpro, lebih baik belajar dahulu Foxbase atau Foxpro.
2. Belajar HTML relatif mudah. Pemula hanya menghapal tanda *tag*  dan kegunaanya, sebagai contoh : <A> (dibaca *tag* A) kegunaannya untuk menghubungkan atau *link* teks/gambar/dokumen <B> kegunaannya untuk teks dicetak tebal, dan seterusnya.
3. Fasilitas untuk mengetik HTML mudah didapat. Di sistem operasi *windows* dapat menggunakan *notepad,* jika anda menggunakan linux dapat menggunakan fasilitas *vi* atau jika anda menggunakan *unix* dengan fasilitas *pico (pine composer).*
4. PHP (*Personal Home Page)*

Menurut Anhar (2010a:3) “PHP : *Hypertext Preprocessor* yaitu bahasa perograman *web server-side* yang bersifat *open source”.* PHP merupakan *script* yang terintegrasi dengan HTML dan berada pada *server (server side* HTML *embedded scripting).* PHP adalah *script* yang digunakan untuk membuat halaman *website* yang dinamis. Dinamis berarti halaman yang akan ditampilkan dibuat saat halaman itu diminta oleh *client.* Mekanisme ini menyebakna informasi yang diterima *client*  selalu yang ter-*up to date.* Semua *script* PHP dieksekusi pada server dimana *script* tersebut dijalankan beberapa alasan bagi kita untuk mempelajari PHP adalah karena :

1. Kesederhanaan, *user* yang baru belajar perograman alasan ini pasti merupakan alasan utama untuk memulai belajar PHP. Karena kesederhanaan tersebut, maka kita menjadi merasa mudah untuk belajar PHP. *User* yang sedikit tahu atau bahkan sama sekali tidak mengerti tentang perrograman PHP bisa dengan cepat belajar dan mencoba membuat aplikasi *web* PHP. Selain itu, PHP memiliki banyak sekali fungsi *built-in* untuk menangani kebutuhan standart pembuatan aplikasi *web.* Dengan adanya fungsi-fungsi tersebut, maka tentu sajaproses belajar PHP terutama dalam pengembangan aplikasi akan jauh lebih mudah karena semua sudah tersedia. Kita sebagai *user*  tinggal memakai dan mengembangkannya.
2. Dalam sisi pengembangan, bahasa PHP adalah *script*  yang paling mudah karena memiliki referensi yang banyak.
3. PHP adalah bahas *open source* yang dapat digunakan diberbagai sistem operasi seperti : Linux, Unix Macintosh, dan Windows. PHP dapat dijalankan secara *runtime* melalui *console* serta dapat menjalankan perintah-perintah *system. Open source* artinya *code-code* PHP terbuka untuk umum dan kita tidak harus membayar biaya pembelian atas keaslian *license* yang biasa cukup mahal. Karena *source code* PHP tersedia secara gratis, maka hal tersebut memungkinkan komunitas milis-milis dan *developer* untuk selalu melakukan perbaikan, pengembangan, dan menemukan *bug* dalam bahasa PHP.
4. *Web Server* yang mendukung PHP dapat ditemukan dimana-mana, mulai dari Apache, IIS, Lighttpd, hingga Xitami dengan *configurasi* yang relatif mudah.
5. PHP jugan dilengkapi dengan berbagai macam pendukung lain seperti *support* langsung ke berbagai macam *database* yang popular. Misalnya : Oracle, PostgreSQL, MySQL, dan lain-lain.
6. Dreamweaver

Menurut Andi (2008:22) : *“Dreamweaver* merupakan program uyang digunakan untuk membuat desain *web* HTML. Dalam program ini juga dapat menyisikan *script-script* pemrograman PHP di sela-sela kode-kode HTML yang ada. *Dreamweaver* bahkan menyediakan fasilitas file baru dalam tipe .php langsung”.

Bagaimana *website* tiba-tiba dapat tampil dihadapan kita? Banyak proses yang berjalan sebelum sebuah *website* dapat dinikmati. Beberapa istilah yang perlu kita ketahui bersama (bagi yang belum familiar tentunya)

Sebelum kita membangun sebuah wesite, diantaranya :

1. *Domain*

*Domain* adalah nama dari alamat *website*  seperti [*www.google.com*](http://www.google.com)*.*

*Domain* seperti ini disebut *Top Domain Level* (TDL)*.* Nah, alamat *website* gratisan yang anda peroleh dari Geocities misalnya merupakan *subdomain.* TDL jelas lebih keren dan lebih profesional di banding *subdomain.* Tentu saja, TDL harus beli. Di dunia ini belum ada yang memberikan *domain* gratis, keculai *domain-domain* yang tidak begitu umum, seperti .co.cc, .tk, dan lainya. Sedangkan untuk TDL yang ada paling menggratiskan *domain* untuk membeli paket layanan tertentu dari perusahaan tersebut, misalnya membeli paket *hosting, domain* harus dibayar per tahun dan setiap tahunnya harus di perbaharui.

Contoh ekstensi domain global adalah .com, .net, .org, sedangkan *domain* denga ekstensi dari indonesia adalah .co.id, .we.id, .or.id, .go.id, .sch,id.

1. *Hosting*

Setelah membeli *domain*, anda perlu meletakan “rumah *domain*” tersebut ke sebuah tempat yang disebut *hosting*. *Hosting* ini tempat meletakan file-file yang akan anda tampilkan di dalam *website*. *Hosting* berupa seperangkat komputer yang disebut *server* yang harus *online* 24 jam agar *website* anda bisa di akses 24 jam oleh pengunjung.

1. *Web Server*

*Web Server* merupakan perangkat lunak yang memungkinkan *script-script*  dalam file *website* berjalan dan diakses serta menyungguhkan tampilan sesuai kehendak kita. Beberapa contoh perangkat lunak *web server* adalah Apache, IIS, Nginx.

1. *Database*

*Database* (Basis Data) merupakan kumpulan data yang saling berhubungan. Terdapat struktur baris dan kolom dalam *database* yang akan menampung *record-record* data *website.* Contoh *Database Management System* yang sering digunakan adalah MySQL, PostGres, Microsoft SQL, ADO DB, SybBase SQL, Oracle.

1. *Web Programming*

Orang-orang yang membangun sebuah *website b*iasanya menggunakan kemampuan pemrograman masing-masing. Terdapat banyak bahasa perograman yang sering dipakai untuk membuat *website,* mulai dari HTML, PHP, ASP, (Microsoft), JSP (Java).

1. CSS (*Cascoding Style Sheets)*

Menurut Sugiri dan Kurniawan (2007:21) “CSS(*Cascoding Style Sheets)* adalah sebuah cara untuk memisahkan isi dengan layout dalam halaman-halaman web yang dibuat”. CSS memperkenalkan *template*  yang berupa *style* untuk membuat dan mempermudah penulisan dari halaman-halaman yang di rancang.

Hal ini sangat penting karena halaman yang menggunakan CSS dapat di baca secara bolak-balik dan isinya dapat dilihat oleh pengunjung dari manapun. CSS mampu menciptakan halaman yang tampak sama pada resolusi layar dari pengunjung yang berbeda tanpa memerlukan suatu *table.* Dengan CSS, Anda akan lebih mudah melakukan *setting* tampilan keseluruhan *web*  hanya dengan mengganti atribut-atribut yang diinginkan tanpa harus mengubah satu persatu atribut tiap elemen yang ada dalam situs yang di buat.

Saat ini CSS merupakan *style* yangbanyak digunakan karena berbagai kemudahan dan kelengkapan atribut yang di milikinya. Penggunaan CSS dalam *web*  akan lebih efisien karena CSS dapat digunakan untuk penggunaan secara berulang pada *tag-tag* tertentu sehingga tidak usah mengetikkan ualng seluruh perintah pemformatan seperti halnya HTML, klasik, CSS akan lebihnmudah dipelajari jika telah mengetahui struktur pembuatan dokumen *web* dengan bahasa HTML.

1. ***Unified Modelling Language (*UML)**

Menurut Munawar (2005a:17) mengemukakan “UML (*Unified Modelling Language)* sebagai salah satu alat bantu yang sangat handal di dunia pengembangan sistem yang berorientasi obyek”. Hal ini disebabkan karena UML menyediakan bahasa pemodelan *visual* yang memungkinkan bagi pengembang sistem untuk membuat cetak biru atas visi mereka dalam bentuk yang baku, mudah dimengerti serta di lengkapi dengan mekanisme yang efektif untuk berbagi (*sharing)* dan mengkomunikasikan rancangan mereka dengan yang lain.

UML merupakan kesatuan dari bahasa pemodelan yang dikembangkan oleh Booch, *Object Modelling Technique* (OMT) dan *Object Oriented Software Engineering* (OOSE). Metode Booch dari Grady Booch sangat terkenal dengan nama metode *Design Object Oriented.*

Abstraksi konsep dasar UML terdiri dari *structural classification, dynamic behavior* dan *model management.* UML mendifinisikan diagram-diagram sebagai berikut :

1. Diagram Use-case (Use Case Diagram)

*Use case* adalah dekripsi fungsi dari sebuah sistem dari perspektif pengguna. *Use case* bekerja dengan cara mendeskripsikan tipikal interaksi antara *user* (pengguna) sebuah sistem dengan sistemnya sendiri melalui sebuah cerita bagaimana sebuah sistem dipakai menurut Munawar (2005:63).

1. Diagram Aktifitas (*Activity Diagram)*

*Activity diagram* adalah teknik untuk mendiskripsikan logika prosedural, proses bisnis dan aliran kerja dalam banyak kasus. *Activity Diagram* mempunyai peran seperti halnya *Flowchart,* akan tetapi perbedaannya dengan *flowchart* adalah *Activity Diagram* bisa mendukung prilaku paralel sedangkan *flowchart* tidak bisa menurut Munawar (2005:109).

1. Diagram Komponen (*Component Diagram)*

Menurut Fowler dalam Munawar (2004:122) mengemukakan “hal penting pada *component adalah component* mewakili potongan-potongan yang independen yang bisa dipesan dan diperbaharui sewaktu-waktu”.

1. Diagram *Deployment* *(Deployment Diagram)*

*Deployment Diagram* menunjukan tata letak sebuah sistem secara fisik, menampakan bagian-bagian *software* yang berjalan pada bagian *hardware.* Bagian utama *hardware* atau perangkat keras adalah node yaitu nama umum untuk semua jenis sumber komputasi. Ada dua tipe node yaitu *processor* dan *device. Processor* adalah node yang bisa mengeksekusi sebuah *component,* sedangkan *device* tidak. *Device* adalah perangkat keras (seperti printer, monitor) tipikalnya menjadi *interface* dengan dunia luar menurut Munawar (2005:125).

Langkah-langkah penggunaan UML secara umum yaitu :

1. Buatlah daftar *business process* dari level tertinggi untuk mendefinisikan aktivitas dan proses yang mungkin muncul.
2. Petakan *use case* untuk tiap *business process* untuk mendefinisikan dengan tepat fungsionalitas yang harus disediakan oleh sistem. Kemudian perhalus *use case diagram* dan lengkapi dengan *recuirment, constraints* dan catatan-catatan lain.
3. Buatlah *deployment diagram* secara kasar untuk mendefinisikan arsitektur fisik sistem.
4. Definisikan *recuirment* lain (non-fungsional, *security* dan sebagainya) yang juga harus disediakan oleh sistem.
5. Berdasarkan *use case diagram,* mulailah membuat *activity diagram.*
6. Definisikan objek-objek level atas (*package* atau *domain)* dan buatlah *squence* atau *calloboration diagram* untuk tiap alir perkerjaan. Jika sebuah *use case* memilki kemungkinan alir normal dan *error,* buatlah satu *diagram* untuk masing-masing alir.
7. Buatlah rancangan *user interface* model yang menyediakan antarmuka bagi pengguna untuk menjalankan skenario *use case.*
8. Berdasarkan model-model yang sudah ada, buatlah *class diagram.* Setiap *package* atau *domain* dipecah menjadi hirarki *class* lengkapdengan atribut dan metodenya. Akan lebih baik jika untuk setiap *class* dibuat *unit test* untuk menguji fungsionalitas *class* dan interaksi dengan *class* lain.
9. Setelah *class diagram* dibuat, kita dapat melihat kemungkinan pengelompokan *class* menjadi komponen-komponen. Karena itu buatlah *component diagram* pada tahap ini. Juga definisikan tes integrasi untuk setiap komponen meyakinkan ia berinteraksi dengan baik.
10. Perhalus *deployment diagram* yang sudah dibuat. Detailkan kemampuan dan *requirment* piranti lunak, sisitem operasi, jaringan, dan sebagainya. Petakan komponen ke dalam node.\
11. Mulailah membangun sistem. Ada dua pendekatan yang dapat digunakan :
12. Pendekatan *use case,* dengan meng-*assign* setiap *use case* kepada tim pengembang tertentu untuk mengembangkan *unit code* yang lengkapdengan tes.
13. Pendekatan komponen, yaitu meng-*assign* setiap komponen kepada tim pengembang tertentu.
14. Lakukan uji modul dan uji integrasi serta perbaiki model beserta *code* nya*.* Model harus selalu sesuai dengan *code* yang aktual.
15. Piranti lunak siap dirilis
16. ***Entity Relationship Diagram* (ERD)**

“Model *Entity Relationship* merupakan suatu model untuk menjelaskan hubungan antar data dalam basis data baerdasarkan suatu persepsi bahwa *real world* terdiri dari *object-object* dasar yang mempunyai hubungan atay relasi antar *object-object* tersebut” Marlinda (2006:4). Model *entity relationship* adalah suatu penyajian data dengan menggunakan *entity* dan *relationship.*

Mengapa di perlukan model E – R :

1. Dapat menggambarkan hubungan antar *entity* dengan jelas.
2. Dapat menggambarkan batasan jumlah *entity* dan partisipasi antar *entity.*
3. Mudah dimengerti oleh pemakai.
4. Mudah disajikan oleh perancang *database.*

Komponen-komponen yang terdapat di dalam *Entity Relationship Model :*

1. Entitas (*Entity)*

*Entity* adalah yang di dapat di bedakan dalam dunia nyata dimana informasi yang berkaitan dengannya dikumpulkan. *Entity set* adalah kumpulan *entity* yang sejenis. Simbol yang digunakan untuk *entity* adalah persegi panjang.

*Entity set* dapat berupa:

1. *Entity* yang bersifat fisik yaitu *entity* yang dapat dilihat

Contohnya: rumah, kendaraan, mahasiswa, dosen dan lain-lain.

1. *Entity* yang bersifat konsep atau *logic* yaitu *entity* yang tidak dapat dilihat. Contohnya: pekerjaan, perusahaan, matakuliah dan lain-lain.
2. *Relatioship*

*Relatioship* adalah hubungan yang terjadi antara satu lebih *entity. Relatioship* tidak mempunyai keberadaan fisik, kecuali yang mewarisi hubungan antara entity tersebut. *Relatioship set*  adalah kumpulan *Relatioship* yang sejenis.

1. *Atribute*

Atribut adalah karakteristik dari *entity* atau *Relatioship* yang menyediakan penjelasan detaik tentang atau *Relatioship* tersebut. *Atribute value* adalah suatu data *actual* atau informasiyang disimpan disuatu *atribute* di dalam suatu *entity* atau *Relatioship.* Terdapat dua jenis *atribute,* yaitu :

1. *Identifer (key),* untuk menentukan suatu *entity* secara unik.
2. *Descriptor (nonkey atribute),* untuk menentukan karakteristik dari suatu *entity* yang tidak unik.
3. *Indicator Tipe*
4. *Indicator Tipe Associative Object*

Berfungsi sebagai objek dan suatu *relationship*

1. *Indicator Tipe Supertipe*

Terdiri dari suatu object dan suatu sub kategori atau lebih yang dihubungkan dengan satu *relationship* yang tidak bernama.

1. *Cardinality Ratio atau Cardinality*

*Cardinality Ratio* adalah menjelaskan hubungan batasan jumlah keterhubungan satu *entity* lainnya atau banyaknya *entity* yang bersesuaian dengan *entity* yang lain melalui *relationship.*

Jenis *Cardinality Ratio* yaitu :

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

Tingkat hubungan dinyatakan satu ke satu jika suatu kejadian pada entitas pertama hanya mempunyai satu hubungan dengan satu kejadian maupun satu hubungan dengan satu kejadian pada entitas kedua. Demikian juga sebaliknya, satu kejadian pada entitas yang kedua hanya bisa mempunyai satu hubungan dengan satu kejadian entitas yang pertama.

1. Satu ke banyak (*one to many* (1:M))

Tingkat hubungan satu kebanyak (1:M) adalah sama dengan banyak ke satu (M:1) tergantung dari arah mana hubungan-hubungan tersebut dilihat. Ntuk satu kejadian pada entitas yang pertama dapat mempunyai banyak hubungan dengan kejadian yang kedua, sebaliknya banyak kejadian pada entitas yang kedua hanya bisa mempunyai satu hubungan dengan satu kejadian pada entitas yang pertama.

1. Banyak ke banyak (*many to many* (M:N))

Tingkat hubungan banyak ke banyak terjadi jika tiap kejadian pada sebuah entitas akan mempunyai hubungan dengan kejadian pada entitas lainnya, baik dilihat dari entitas yang pertama maupun dilihat dari entitas yang kedua.

Derajat *relationship* menyatakan jumlah *entity* yang berpatisipasi di dalam suatu *relationship* yaitu :

1. *Unary egree* (derajat satu) adalah derajat yang memiliki satu *relationship* untuk satu *entity.*
2. *Binary Degree* (derajat dua) adalah derajat yang memilki satu *relationship* untuk dua buah *entity.*
3. *Ternary Degree* (derajat tiga) adalah derajat yang memilki satu *relationship* untuk tiga buah *entity.*
4. *Participation Constraint*

Menjelaskan apakah keberadaan suatu *entity* tergantung pada hubungannya dengan *entity* lain.

Terdapat dua macam *Participation Constraint,* yaitu:

1. *Total Participation,* yaitu keberadaan suatu *entity* tergantung pada hubungannya dengan *enity* lain. Di dalam diagram E-R di gambarkan dengan dua garis penghubung antar *entity* dan *relationship.*
2. *Parttial Participation,* yaitu keberadaan suatu *entity* tidak tergantung pada hubungan dengan *entity* lain. Di dalam diagram E-R di gambarkan dengan dua garis penghubung antar *entity* dan *relationship.*
3. Representasi dari *entity set*

*Entity Set* di representasikan dalam bentuk *table* dan nama yang *unique.* Setiap *table* terdiri dari sejumlah kolom.Masing-masing kolom diberi nama yang *unique* pula*.*

*Entity set* terbagi atas:

1. *Strong entity set, entity set* yang satu atau lebih *atribute*nya digunakan oleh *entity set* lain sebagai *key* digambarkan dengan empat persegi panjang.
2. *Weak entity set, entity set dependent* terhadap *strong entity set.* Keberadaan *entity* tersebut keberadaan *entity* lain. *Entity* lain tersebut disebut *identifying owner* dan *relationship* disebut *identifying relationship. Weak entity* selalu mempunyai total *participation constraint* dengan *identifying owner. Weak entity* digambarkan dengan persegi panjang bertumpuk.

**2.2 Penelitian terkait**

Handita dkk (2011:1) “Penggunaan E-Commerce bagi pihak produsen dapat membantu meningkatkan kemajuan perusahaan, seperti hal pemasaran, pemesanan, sampai proses transaksi penjualan. Perkembangan dunia bisnis yang semakin maju dan tingkat persaingan yang bertambah besar, membangkitkan kesadaran para pengusaha untuk bekerja lebih keras dalam mengelola perusahaannya. Perusahaan yang ingin meningkatkan penjualan tidak dapat melepaskan diri dari kegiatan pemasaran. Pemasaran yang baik dan yang tersebar luas merupakan faktor untuk meningkatkan hasil penjualan perusahaan”.

Himawan dkk (2014:4) “Metodologi Pengembang sistem dapat berarti menyusun sistem yang baru untuk menggantikan sistem yang lama secara keseluruhan atau memperbaiki sistem yang telah ada. Sistem yang lama perlu diperbaiki karena adanya permasalahan-permasalahan yang timbul di sistem yang lama”.