BAB II

TINJAUAN PUSTAKA DAN LANDASAN TEORI

2.1. Tinjauan Pustaka

Tinjauan pustaka dilakukan berdasarkan pada penelitian terdahulu, maka dari itu dapat dipaparkan sebagai berikut:

Menurut Dendi Permana Putra dari Perguruan Tinggi Bina Darma, Palembang tahun 2013 dalam penelitiannya yang berjudul "Analisis dan Perancangan Basis Data Penjualan, Pembelian dan Persediaan Barang pada CV. Cemerlang Jaya" merupakan perancangan *database* yang berfungsi untuk membantu perusahaan CV. Cemerlang Jaya dalam menciptakan basis data yang mampu mengorganisir data penjualan, pembelian, persediaan barang dan data-data lainnya menjadi suatu kumpulan data yang terhubung (*interrelated data*) yang disimpan secara bersama-sama pada suatu media sehingga mudah digunakan atau dimanfaatkan kembali. Tujuan dilakukannya penelitian ini adalah melakukan analisis dan perancangan basis data mulai dari penjualan, pembelian, persediaan pada CV. Cemerlang Jaya, yang diharapkan akan mempercepat dan mempermudah pendataan penjualan, pembelian dan persediaan barang pada CV. Cemerlang Jaya. Metode perancangan basis data yang digunakan adalah metode analisis desain representasi fisik dan metode perancangan DBLC (*Database Life Cycle*).

Penelitian yang dilakukan oleh Sudaryadi dari Perguruan Tinggi Bina Darma, Palembang pada tahun 2013 yang berjudul "Analisis dan Perancangan Basis Data Inventaris Dinas Perkebunan Kabupaten Ogan Komering Ilir". merupakan perancangan *database* inventaris yang mampu mengelola barang-barang inventaris yang terdapat di Dinas Perkebunan Kabupaten Ogan Komering Ilir sehingga bisa terkoordinasi dengan baik. Basis data ini tidak hanya sebagai *database* untuk menyimpan data namun mampu untuk menghitung jumlah setiap barang yang masuk, keluar, ataupun yang sudah rusak, serta adanya pelaporan untuk penggunaan barang inventaris oleh pegawai. Tujuan dilakukannya penelitian ini adalah menganalisis kebutuhan dan merancang basis data inventaris untuk mengolah data inventaris pada Dinas Perkebunan Kabupaten Ogan Komering Ilir. Metode perancangan basis data yang digunakan adalah *Database Lifecycle* (DBLC) yang terdiri dari *Database Planning* (Perencanaan Basis data), *System definition* (definisi sistem), *Requirement Collection and Analysis* (Pengumpulan dan Analisa Data), Perancangan *Database* Konseptual, Perancangan *Database* Logikal, Perancangan *Database* Fisikal.

Ita Rosita Wati dari (2013) dengan judul "Rancangan dan Implementasi Basis Data Perpustakaan (Study Kasus SMK Panggali Nusantara Palembang)". merupakan perancangan *database* pada perpustakaan. Metode perancangan basis data yang digunakan adalah *Database Life Cycle* (DBLC) yang terdiri dari *Database Planning* (Perencanaan Basis data), *System definition* (definisi sistem), *Requirement Collection*

and Analysis (Pengumpulan dan Analisa Data), Perancangan *Database* Konseptual, Perancangan *Database* Logikal, dan Perancangan *Database* Fisikal.

Dalam pembuatan skripsi ini tahapan yang digunakan dalam perancangan *database* adalah *Requirement Collection and Analysis* (Pengumpulan dan Analisa Data), *Conceptual Database Design* (Perancangan *Database* Konseptual), *Logical Database Design* (Perancangan *Database* Logikal), dan *Physical Database Design* (Perancangan *Database* Fisikal). Selain merancang *database*, penelitian ini juga melakukan pengujian pada *database* apakah *database* mengalami anomali atau tidak, dengan dilakukan pengujian dapat diperoleh *database* dengan data yang bagus dan efisien.

2.2. Landasan Teori

2.2.1. *Database*

Sistem pemrosesan basis data terbentuk setelah masa sistem pemrosesan manual dan sistem pemrosesan berkas. Sistem pemrosesan manual (berbasis kertas) merupakan bentuk pemrosesan yang menggunakan dasar berupa setumpuk rekaman yang disimpan pada rak-rak berkas. Maka dapat disimpulkan basis data adalah sistem berkas terpadu yang dirancang terutama untuk meminimalkan pengulangan data (Kadir, 2003).

Basis data adalah kumpulan data yang saling berelasi. Data sendiri merupakan fakta mengenai obyek, orang, dan lain-lain. Data dinyatakan dengan nilai (angka,

deretan karakter, atau *symbol*). Basis data bertujuan untuk mengatur data sehingga diperoleh kemudahan, ketepatan, dan kecepatan dalam pengambilan kembali.

Basis data dapat didefinisikan dalam berbagai sudut pandang seperti berikut:

1. Himpunan kelompok data yang saling berhubungan yang diorganisasi sedemikian rupa sehingga kelak dapat dimanfaatkan dengan cepat dan mudah.
2. Kumpulan data yang saling berhubungan yang disimpan secara bersama sedemikian rupa tanpa pengulangan (*redudancy*) yang tidak perlu, untuk memenuhi kebutuhan.
3. Kumpulan *file* atau tabel atau arsip yang saling berhubungan yang disimpan dalam media penyimpan elektronik (Kusrini, 2007).

2.2.2. *Database Management System* (DBMS)

Sistem manajemen *database* adalah sistem berbasis komputer untuk mendefinisikan, membuat, memanipulasi, mengawasi, mengatur, dan menggunakan *database*. Sebuah *database* adalah kumpulan dari integrasi data yang terorganisir, seperti *byte*, ruas, *record*, dan *file*. Penerapan basis data dalam sistem informasi disebut *Database Management System* (DBMS) yang memungkinkan untuk penyimpanan data, pemeliharaan data, pengelolaan akses, keamanan, dan pengintegrasian menjadi bahan pendukung keputusan (*Decission Support System*). DBMS menyediakan fasilitas untuk mengedit, menambahkan, menghapus, menampilkan, mencari, memilih, mengurutkan, dan mencetak data (Supriyanto, 2008).

Database Management System (DBMS) merupakan paket program (*software*) yang dibuat agar memudahkan dan mengifisienkan pemasukan, pengeditan, penghapusan, dan pengmabilan informasi terhadap *database*. *Software* yang tergolong kedalam DBMS antara lain, *Microsoft SQL*, *MySQL, Oracle, Ms. Access*, dan lain-lain (Yanto, 2016).

2.2.3. *Relational Database*

Basis Data relasional adalah basis data yang menyimpan data dalam tabel-tabel yang terdiri atas baris dan kolom. Setiap baris memiliki *primary key* dan setiap kolom memiliki nama tersendiri. Basis data relasional menggunakan istilah-istilah yang berbeda dengan sistem pemrosesan *file*. Seorang pengembang basis data relasional merujuk pada *file* sebagai suatu relasi (*relation*), *record* sebagai *tuple*. dan *field* sebagai atribut (*attribute*). Seorang pengguna basis data relasional, sebaliknya merujuk pada *file* sebagai tabel (*table*), *record* sebagai baris (*row),* dan *field* sebagai kolom (*column*). Selain menyimpan data, basis data relasional juga menyimpan hubungan-hubungan antar data. Dalam basis data relasional anda dapat menentukan relasi antara tabel-tabel pada suatu waktu (Shelly, Cashman, & Vermaat, 2007).

Desain basis data relational dibagi atas 3 fase yaitu:

1. *Conceptual Database Design*

   Merupakan suatu proses pembentukan model yang berasal dari informasi yang digunakan dalam perusahaan yang bersifat independen dari keseluruhan aspek

fisik. Model data tersebut dibangun menggunakan informasi dalam spesifikasi kebutuhan *user* dan merupakan sumber informasi untuk fase desain *logical*.

2. *Logical Database Design*

   Merupakan suatu proses pembentukan model yang berasal dari informasi yang digunakan dalam perusahaan yang berdasarkan model data tertentu, namun independen terhadap DBMS tertentu dan aspek fisik lainnya, misalnya model data konseptual yang telah dibuat sebelumnya, diperbaiki dan dipetakan kembali ke dalam model data *logical*.

3. *Physical Database Design*

   Merupakan proses yang menghasilkan deskripsi implementasi basis data pada penyimpanan sekunder, menggambarkan struktur penyimpanan dan metode akses yang digunakan untuk mencapai akses yang efisien terhadap data. Dapat juga dikatakan bahwa desain fisikal merupakan cara pembuatan menuju DBMS tertentu (Yanto, 2016).

2.2.4. *Entity Relationship Diagram* (ERD)

Entity Relationship Diagram adalah suatu model jaringan yang menggunakan susunan data yang disimpan dalam sistem secara abstrak. ERD merupakan model jaringan data yang menekankan pada struktur-struktur dan *relationship* data (Sutanta, 2004).

Entity Relation Diagram (ERD) adalah notasi yang digunakan untuk melakukan aktivitas pemodelan data. Tujuan utama dari *Entity Relationship Diagram* (ERD) adalah mewakili objek data dan hubungan mereka (Fathansyah, 2007).

Komponen utama identifikasi untuk *Entity Relationsip Diagram* (ERD) berupa:

1. Entitas (*Entity*)

   Entitas adalah suatu objek di dunia nyata yang dapat dibedakan dengan objek lainnya. Objek tersebut dapat berupa orang, benda ataupun hal lainnya.

   Entitas digambarkan dalam bentuk persegi panjang seperti pada tabel 2.1, dilihat dari jenisnya entitas terbagi atas dua yaitu:

   a. Entitas Kuat (*Strong Entity*)

      Entitas kuat adalah entitas yang dapat berdiri sendiri tidak tergantung pada entitas lainnya. Contoh entitas kuat adalah entitas pegawai.

   b. Entitas Lemah (*Weak Entity*)

      Entitas lemah adalah entitas yang tidak dapat berdiri sendiri. Entitas lemah merupakan hasil dari entitas kuat. Contoh entitas lemah adalah entitas pegawai kontrak, pegawai tetap.

2. Atribut (*Attribute*)

   Atribut merupakan semua informasi yang berkaitan dengan entitas. Atribut sering dikenal dengan *property* dari suatu entitas atau objek. Atribut digambarkan dalam bentuk lingkaran elips seperti pada tabel 2.1. Macam-macam atribut yaitu:

   a. Atribut Sederhana (*Simple Attribute*)

Atribut sederhana adalah atribut yang nilainya tidak dapat dibagi lagi menjadi banyak atribut yang lebih kecil. Contoh atribut sederhana yaitu harga.

b. Atribut Komposit (*Composite Attribute*)

Atribut komposit adalah atribut gabungan yang nilainya dapat dipecah menjadi bagian yang lebih kecil atau sering disebut atribut yang terdiri dari beberapa atribut kecil didalamnya. Contoh atribut komposit adalah alamat, alamat dapat dipecah lagi yaitu kota, provinsi, kode_pos, dan no-jln.

c. Atribut Bernilai Tunggal (*Single Values Attribute*)

Atribut bernilai tunggal adalah jenis atribut yang nilainya hanya satu dari suatu entitas. Contoh atribut bernilai tunggal adalah tanggal_lahir dari entitas mahasiswa, telah bisa dipastikan bahwa setiap mahasiswa mepunyai satu tanggal_lahir.

d. Atribut Bernilai Banyak (*Multivalues Attribute*)

Atribut bernilai banyak adalah jenis atribut yang nilainya lebih dari satu dalam entitas tertentu. Contoh atribut bernilai banyak adalah hobbi, dimungkinkan bahwa mahasiswa meiliki lebih dari satu hobbi.

e. Atribut Turunan (*Derived Attribute*)

Atribut turunan adalah jenis atribut yang nilainya diperoleh dari atribut yang lain. Contoh atribut adalah masa_bakti dari entitas pegawai. Atribut masa_bakti akan muncul nilainya ketika atribut tanggal_masuk_kerja sudah ada nilainya.

f. Atribut Identitas (*Key Attribute*)

Atribut identitas adalah atribut yang dijadikan sebagai kunci pada suatu tabel. Sifat atribut identitas ini unik, tidak ada yang menyamai, atribut identitas terdiri dari beberapa jenis yaitu:

1. *Super Key*

   Super key adalah satu atribut atau kumpulan atribut yang secara unik mengidentifikasi sebuah baris di dalam relasi atau himpunan dari satu atau lebih entitas yang dapat digunakan untuk mengidentifikasi secara unik sebuah entitas dalam set entitas.

2. *Candidate Key*

   Candidate key adalah atribut yang menjadi determinan yang dapat dijadikan identitas baris pada sebuah relasi.

3. *Primary Key*

   Primary key adalah kandidat *key* yang dipilih untuk mengidentifikasi baris data secara unik dalam relasi.

4. *Alternative Key*

   Alternative key adalah *candidate key* yang tidak terpilih sebagai *primary key* atau atribut untuk menggantikan kunci utama.

5. *Foreign Key*

   Foreign key adalah atribut dengan domain yang sama yang menjadi kunci utama sebuah relasi, tetapi pada relasi lain atribut sebagai atribut biasa.

6. *Composite Key*

   Composite key adalah kunci yang terdiri dari dua atribut atau lebih. Atribut-atribut tersebut jika berdiri sendiri tidak menjadi identitas baris, tetapi bila dirangkaikan menjadi satu kesatuan akan dapat mengidentifikasikan secara unik.

3. Relasi

Hubungan ditunjukkan dengan garis yang diberi label yang menghubungkan objek. Sambungan antara objek dan hubungan dibangun dengan menggunakan kardinalitas dan modalitas. Relasi digambarkan dalam bentuk belah ketupat seperti pada tabel 2.1. Derajat kardinalitas merupakan penjabaran dari hubungan antar entitas.

Derajat kardinalitas dibagi atas 3 bagian yaitu:

a. Derajat kardinalitas *One to One*

Derajat kardinalitas *one to one* terjadi jika satu entitas X hanya berelasi dengan satu entitas Y, ataupun sebaliknya. Sebagai contoh satu pegawai studi hanya memiliki satu pendamping.

b. Derajat kardinalitas *One to Many*

Derajat kardinalitas *one to many* terjadi jika satu entitas X berelasi dengan banyak entitas Y, ataupun sebaliknya. Sebagai contoh satu dosen mengampu banyak mahasiswa.

c. Derajat kardinalitas *Many to Many*

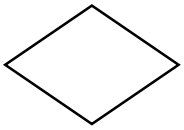
Derajat kardinalitas *many to many* terjadi jika banyak entitas X berelasi dengan banyak entitas Y, ataupun sebaliknya. Sebagai contoh banyak mahasiswa mengambil banyak matakuliah.

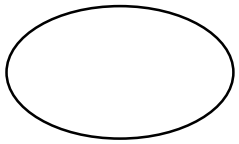
4. Garis

Adalah tanda garis yang digunakan untuk menghubungkan komponen-komponen ERD (Yanto, 2016). Garis dapat dilihat pada tabel 2.1.

Adapun simbol-simbol dari *Entity Relationship Diagram* (ERD) adalah sebagai berikut:

Tabel 2.1 Simbol-simbol dalam ER Diagram

SIMBOL	NAMA	KETERANGAN
	Entitas	Digambarkan dalam bentuk persegi panjang. Entitas adalah sesuatu apa saja yang ada dalam sistem, nyata maupun abstrak dimana data disimpan atau dimana terdapat data.
	Relasi	Relasi adalah hubungan ilmiah yang terjadi antara entitas.

SIMBOL	NAMA	KETERANGAN
	Atribut	Adalah sifat atau karakteristik dari tiap-tiap entitas dan relasi atau elemen data dari entitas dan relasi. Atribut ini digunakan untuk penamaan dari bagian-bagian yang terdapat dalam entitas.
	Garis	Menghubungkan antara entitas satu dengan entitas lainnya.

2.2.5. *MySQL*

MySQL adalah salah satu jenis *database* yang terkenal dan termasuk jenis RDBMS (*Relational Database Management System*). Kepopuleran *MySQL* disebabkan karena *MySQL* menggunakan bahasa SQL sebagai bahasa dasar untuk *query* dan bersifat *open sources* di berbagai *platform* (Kadir, 2008:348).

MySQL adalah *Relational Database Management System* (RDBMS) yang didistribusikan secara gratis di bawah lisensi GPL (*General Public License*). Dimana setiap orang bebas untuk menggunakan *MySQL*, namun tidak boleh dijadikan produk turunan yang bersifat *closed source* atau komersial. *MySQL* sebenarnya merupakan turunan salah satu konsep utama dalam *database* sejak lama, yaitu SQL (*Structured Query Language*). SQL adalah sebuah konsep pengoperasian *database*, terutama untuk pemilihan atau seleksi dan pemasukan data yang memungkinkan pengoperasian data dikerjakan dengan mudah secara otomatis. Sebagai *database server*, *MySQL* dapat

dikatakan lebih unggul dibandingkan *database server* lainnya, terutama dalam kecepatan. Berikut beberapa keistimewaan *MySQL,* antara lain:

1. *Portability*

   MySQL dapat berjalan stabil pada berbagai sistem operasi seperti Windows, Linux, FreeBSD, Mac Os X Server, Solaris Amiga, dan masih banyak lagi.

2. *Multiuser*

   MySQL dapat digunakan oleh beberapa *user* dalam waktu yang bersamaan tanpa mengalami masalah atau konflik.

3. *Security*

   MySQL memiliki beberapa lapisan sekuritas seperti *level subnetmask*, nama *host*, dan izin akses *user* dengan sistem perizinan yang mendetail serta *password* terenkripsi.

4. *Scalability* dan *limits*

   MySQL mampu menangani *database* dalam skala besar, dengan jumlah *records* lebih dari 50 juta dan 60 ribu tabel serta 5 milyar baris. Selain itu batas indeks yang dapat ditampung mencapai 32 indeks pada setiap tabelnya (Huda, 2010).

2.2.6. *PHPMyAdmin*

PHPMyAdmin adalah sebuah *software* berbasis pemrograman PHP yang dipergunakan sebagai administrator *MySQL* melalui *browser* (*web*) yang digunakan untuk manajemen *database*. *PHPMyAdmin* mendukung berbagai aktivitas *MySQL* seperti pengelolaan data, *table*, relasi antar *table,* dan lain sebagainya (Rahman, 2013).

PHPMyAdmin adalah *MySQL client* yang berupa aplikasi *web* dan umumnya tersedia di *server* PHP seperti XAMPP maupun servel komersial lainnya (Zaki & Community, 2008).

2.2.7. XAMPP

XAMPP adalah sebuah *software* yang berfungsi untuk menjalankan *website* berbasis PHP dan menggunakan pengolah data *MySQL* dikomputer *local*. XAMPP berperan sebagai *server web* pada komputer anda. XAMPP juga dapat disebut sebuah CPanel *server* virtual, yang dapat membantu anda melakukan *preview* sehingga dapat memodifikasi *website* tanpa harus *online* atau terakses dengan internet (Wicaksono & Community, 2008).

2.2.8. *Store Procedure*

Store procedure adalah salah satu objek *routine* yang tersimpan pada *database* SQL dan dapat digunakan untuk menggantikan berbagai kumpulan perintah yang sering kita gunakan, seperti misalkan sejumlah *row* ke tabel lain dengan filter tertentu. *Store procedure* sangat berguna ketika kita tidak ingin *user* mengakses tabel secara langsung, atau dengan kata lain membatasi hak akses *user* dan mencatat operasi yang dilakukan. Dengan demikian resiko kebocoran dan kerusakan data dapat lebih diminimalisir (Risma, 2015).

Store procedure ditulis dalam bentuk suatu *Script*. Beberapa keuntungan dari *store procedure* antara lain sebagai berikut:

1. Cepat, kompilasi dilakukan di *Database* (kadang disebut"*pre-compilation*") sehingga mengurangi *traffic*
2. Adanya pemisahan antara *database access logic* dengan *application logic* sehingga program aplikasi menjadi lebih sederhana dan lebih ringkas (*thin client concept*)
3. Berupa obyek dalam *database*, sehingga menghilangkan ketergantungan terhadap bahasa program yang digunakan
4. Bersifat *Portable*, jika bisa berjalan di *database* tersebut maka dipastikan jika *database* bisa terinstal dimanapun maka *store procedure* pasti bisa dijalankan

Adapun sintak untuk membuat *procedure* sebagai berikut:

CREATE PROCEDURE sp_name ([proc_parameter [, …]])

[cha racteristic ..]

routine_body

Stored procedure dapat menggunakan parameter sehingga program aplikasi yang memanggil *stored procedure* dapat mengakses *database* yang diperlukan sesuai dengan kondisi yang diminta yaitu dengan cara mengirimkan suatu nilai ke *stored procedure* melalui parameter. Terdapat 3 mode parameter yaitu :

- *IN (default)* akan mempassingkan nilai konstan dari memori ke *stored procedure*
- *OUT* akan mengambil nilai dari *procedure*
- *IN OUT* akan mempassingkan nilai dari memori ke dalam *procedure* dan memungkinkan nilai yang berbeda dari *procedure* dikembalikan ke memori dengan menggunakan parameter yang sama.