BAB II
TINJAUAN PUSTAKA DAN LANDASAN TEORI

2.1 Tinjaun Pustaka

Sebagai suatu perbandingan dan sumber referensi dalam pengembangan Perancangan Basis Data pada Aplikasi Pariwisata JogjaKuy Berbasis *Android*, diperlukan suatu acuan terhadap penelitian yang dibuat sebelumnya. Adapun hasil penelitian sejenis yang dijadikan referensi adalah sebagai berikut:

Analisis dan Perancangan Basis Data Eksplorasi Berbasis Objek Studi Kasus Kondur Petrolium SA.Penelitian ini dilakukan oleh Suparto Darudiato, mahasiswa Jurusan Sistem Informasi Universitas Bina Nusantara menjelaskan bahwa Kondur Petroleum SA merupakan perusahaan yang bergerak di bidang minyak dan gas. Sistem basis data dapat mengorganisir kegiatan eksplorasi untuk dapat membantu meningkatkan kinerja perusahaan dan mendukung kegiatan eksplorasi perusahaan. Metodologi yang digunakan ada tiga yaitu studi pustaka, penelitian laboratorium, *fact-finding* dengan cara analisa sistem berjalan, survey perusahaan dan wawancara dengan orang yang berhubungan dengan kegiatan eksplorasi. Dengan perancangan basis data yang benar dan baik akan membuat basis data tersebut menjadi fleksibel. *Basis data* yang mudah di *maintain* untuk menghadapi permasalahan yang terus berkembang di masa mendatang.

Menurut Wiharja, 2011, dalam penelitiannya yang berjudul "Apikasi Rental Motor Menggunakan *Java MySQL* dan *JasperReports*". Penulisan ilmiahnya berisi tentang pembuatan sebuah aplikasi rental khususnya rental motor. Di dalam aplikasi

ini terdapat beberapa menu seperti registrasi bagi pelanggan (member) baru,penyewaan, pengembalian motor hingga laporan transaksi. Tujuan dari pembuatan aplikasi rental motor ini adalah untuk membantu memberikan kemudahan dalam pendataan maupun pengelolaan bagi berbagai perusahaan yang bergerak di bidang penyewaan motor. Di dalam pembuatan aplikasi rental motor ini terdiri dari beberapa tahapan yakni pengumpulan data, perancangan tampilan aplikasi, serta tahap uji coba. Pembuatan aplikasi ini menggunakan bahasa pemrograman *J2SE* dengan editor netbeans, *MySQL* sebagai sistem *database*-nya dan *JasperReport* untuk membuat laporannya.

Penelitian tersebut memang sebagian tidak sama dengan judul yang diambil yaitu tentang Perancangan *Basis Data* pada aplikasi JogjaKuy berbasis *Android*, karena masih jarang orang yang melakukan penelitian tersebut. Namun dari itu semua dapat menjadi bahan acuan dan masukan guna ketepatan pelaksanaan sistem dalam penulisan ilmiah, karena masih menggunakan sistem yang sama yaitu pengembangan dengan *Java* dan *MySQL*.

2.2 Landasan Teori

2.2.1 Perancangan

Perancangan adalah sebuah proses untuk mendifinisikan sesuatu yang akan di kerjakan dengan menggunakan teknik yang bervariasi serta di dalamnya melibatkan deskripsi mengenai arsitektur serta detail komponen dan juga keterbatasan yang akan di alami dalam proses pengerjaan (Soetam.2011: 140).

Tahapan perancangan basis data secara umum terdiri dari tiga fase antara lain (Abdul Kadir, 2008:24-28):

2.2.2 Perancangan *Konseptual*

Perancangan basis data konseptual adalah proses membangun model informasi yang digunakan di dalam *database*, bebas dari semua pertimbangan fisikal. Menurut Connolly dan Begg (1998), perancangan konseptual basis data adalah proses membangun sebuah model dari informasi yang digunakan oleh *database*

2.2.3 Perancangan *Logikal*

Perancanan basis data logikal adalah proses membangun model informasi yang dibangun *databse* berdasarkan dari beberapa model data yang spesifik, tetapi bebas dari fakta DBMS dan pertimbangan fisikal lainnya. Menurut Connolly dan Begg (1998), perancangan logikal basis data merupakan proses membangun sebuah model informasi yang digunakan dalam sebuah *database* berdasarkan pada sebuah model data yang spesifik, tetapi tidak bergantung pada sebuah DBMS tertentu dan pertimbangan-pertimbangan fisik lainnya

2.2.4 Perancangan *Fisikal*

Menurut Connolly dan Begg (2010, 523), perancangan fisikal basis data adalah proses menghasilkan deskripsi dari pengimplementasian basis data ke dalam tempat penyimpanan sekunder. Proses ini mendeskripsikan relasi dasar, organisasi file, dan index yang digunakan untuk mencapai keefisienan dalam mengakses data, dan setiap *int*egritas data terkait, dan langkah – langkah keamanan.

2.3 Basis Data

Database merupakan kumpulan informasi yang disimpan dalam sebuah komputer secara sistematik sehingga dapat diperiksa menggunakan suatu komputer untuk memperoleh informasi dari basis data (*database*) tersebut (Fathansyah. 2012).

Penerapan *database* dalam suatu informasi disebut dengan *database System*. *Database* digunakan untuk menyimpan informasi atau data yang ter*int*egrasi dengan baik di dalam komputer.

Database adalah sekumpulan data yang sudah disusun sedemikan rupa dengan ketentuan atau aturan tertentu yang saling berelasi sehingga memudahkan pengguna dalam mengelolanya juga memudahkan memperoleh informasi. Selain itu ada pula yang mendefinisikan *database* sebagai kumpulan *file*, tabel, atau arsip yang saling terhubung yang disimpan dalam media elektronik.

Database terbentuk dari sekelompok data data yang memiliki jenis atau sifat yang sama. Sebagai contoh data nama, data kelas, data alamat dikelompokkan dalam data baru yaitu mahasiswa. Demikian juga, kumpulan dari datadata mahasiswa, datadata dosen, datadata keuangan dan lainnya dapat dikumpulkan lagi menjadi kelompok besar, misalkan datadata jurusan atau fakultas pada sebuah universitas. Bahkan dalam perkembangannya, datadata tersebut dapat berbentuk berbagai macam data, misalkan dapat berupa program,lembaran lembaran untuk *entry* (memasukkan) data, laporanlaporan.Semuanya itu dapat dikumpulkan menjadi satu yang disebut dengan *database*.

Database perlu disimpan di dalam basis data untuk keperluan penyediaan informasi lebih lanjut. Data didalam basis data supaya diorganisasikan sedemikian rupa, sehingga membentuk informasi yang lebih berkualitas.

Perancangan basis data merupakan upaya untuk membangun sebuah *database* dalam suatu lingkungan bisnis. Untuk membangun sebuah basis data terdapat tahapan tahapanyang perlu dilalui yaitu:

1. Perencanaan basis data
2. Mendefinisikan sistem
3. Analisa dan mengumpulkan kebutuhan
4. Perancangan basis data
5. Perancangan aplikasi
6. Membuat prototipe
7. Implementasi
8. Konversi data
9. Pengujian
10. Pemeliharaan operasional

2.4 *Database Management System* (DBMS)

Menurut Connolly and Begg (2005, p16), *Database Management System* (DBMS) adalah sebuah sistem *software* yang memungkinkan pengguna untuk mendefinisikan, membuat, menjaga, dan mengontrol akses ke *database.* Biasanya, sebuah DBMS menyediakan fasilitas – fasilitas sebagai berikut :

1. Fasilitas yang mengizinkan pengguna untuk mendefinisikan *database*, biasanya dengan menggunakan *Data Definition Language* (DDL). DDL

mengizinkan pengguna untuk menentukan tipe data dan strukturnya, serta batasan aturan mengenai data yang akan disimpan ke dalam *database.*

2. Fasilitas yang mengizinkan pengguna untuk *Insert* (memasukkan), *Update* (memperbaharui), *Delete* (menghapus), dan *Retrieve* (memperoleh kembali) data dari *database*, biasanya dengan menggunakan *Data Manipulation Language* (DML). Ada juga fasilitas yang melayani pengaksesan data yang disebut *query language.* Bahasa yang paling umum digunakan adalah *Structured Query Languange* (SQL). Fasilitas yang menyediakan akses kontrol ke dalam *database,* yang meliputi:
    a. *Security system*, mencegah pengguna yang tidak memiliki hak akses untuk memasuki *database.*
    b. *Integrity system*, menjaga konsistensi data yang tersimpan.
    c. *Concurrency control system*, mengizinkan akses ke *database* secara bersama.
    d. *Recovery control system*, memperbaiki dan mengembalikan *database* ke dalam kondisi semula apabila terjadi kerusakan pada *hardware* atau *software.*
    e. *User-accessible catalog*, berisi deskripsi data pada *database.*

2.5 Model Basis Data

Abdul Kadir (2003:47) mengemukakan bahwa, model basis data relasional memiliki beberapa definisi penting sebagai berikut:

1. Kumpulan objek atau relasi untuk menyimpan data

2. Kumpulan dari operator yang melakukan suatu aksi terhadap suatu relasi untuk menghasilkan relasi-relasi lain
3. Basis data relasional harus mendukung *int*egritas data sehingga data tersebut harus akurat dan konsisten.

Basis data relasional memiliki fungsi-fungsi kegunaan sebagai berikut:

1. Mengatur penyimpanan data
2. Mengontrol akses terhadap data
3. Mendukung proses menampilkan dan memanipulasi data. Beberapa istilah yang perlu kita pahami mengenai basis data relasional antara lain:

a. Tabel : Merupakan struktur penyimpanan dasar dari basis data relasional, terdiri dari satu atau lebih kolom (column) dan nol atau lebih baris (row).

b. ***Row*** **(baris)** : Baris merupakan kombinasi dari nilai-nilai kolom dalam tabel; sebagai contoh, informasi tentang suatu departemen pada tabel Departmen. Baris seringkali disebut dengan "record".

c. *Column* (kolom) : Kolom menggambarkan jenis data pada tabel; sebagai contoh, nama departemen dalam tabel Departmen. Kolom di definisikan dengan *nama kolom* dan *tipe data* beserta *panjang data* tertentu.

d. ***Field*** **:** *Field* merupakan pertemuan antara *baris* dan *kolom*. Sebuah field dapat berisi data. Jika pada suatu field tidak terdapat data, maka field tersebut dikatakan memiliki nilai "null".

e. ***Primary key*** **:** *Primary key* atau kunci utama merupakan *kolom* atau *kumpulan kolom* yang secara unik membedakan antara baris yang satu dengan lainnya; sebagai contoh adalah kode departemen. Kolom dengan kategori ini tidak boleh

mengandung nilai *"null"*, dan nilainya harus bersifat *unique* (berbeda antara baris satu dengan lainnya). Perbedaan *Primary key* dengan *UNIQUE* adalah jika suatu baris di *set* sebagai *primary key* maka data *record* pada baris tersebut tidak diperbolehkan *NULL* sedangkan pada *UNIQUE* data *record* pada baris yang di *set UNIQUE* dapat berupa *NULL*

f. ***Foreign key* :** *Foreign key* atau kunci tamu merupakan *kolom* atau *kumpulan kolom* yang mengacu ke *primary key* pada tabel yang sama atau tabel lain. *Foreign key* ini dibuat untuk memaksakan aturan-aturan relasi pada basis data. Nilai data dari *foreign key* harus sesuai dengan nilai data pada kolom dari tabel yang diacunya atau bernilai "null".

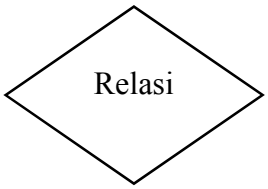
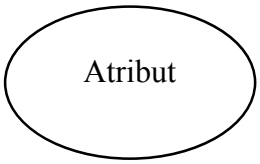
2.7 *Entitiy Relationship Diagram*

Menurut Connolly (2002:330) *Entity Relationship Diagram* digunakan untuk menggambarkan struktur *logical database* dalam bentuk diagram. ERD menyediakan cara yang sederhana dan mudah untuk memahami berbagai komponen dalam desain *database*.

Berikut adalah simbol-simbol khusus yang digunakan untuk menggambarkan elemen-elemen ERD :

Tabel 2.1 ER Diagram

Notasi	Keterangan
Entitas	**Entitas,** adalah suatu objek yang dapat diidentifikasi dalam lingkungan pemakai

Notasi	Keterangan
	**Relasi,** menunjukkan adanya hubungan diantara sejunlah entitas yang berbeda.
	**Atribut,** berfungsi mendeskripsikan karakter entitas (atribut yang berfungsi sebagai *key* diberi garis bawah)
	Garis, sebagai penghubung antara relasi dengan entitas, relasi dan entitas dengan atribut.

2.8 Normalisasi

Normalisasi adalah proses untuk mengubah suatu relasi yang memiliki masalah tertentu kedalam dua buah relasi atau lebih yang tak memiliki masalah tersebut (Kadir, 2007:65). Bentuk normalisasi adalah suatu aturan yang dikenakan pada tabel-tabel dalam basis data dan harus dipenuhi oleh tabel-tabel tersebut pada level-level normalisasi. Ada macam-macam bentuk normalisasi, diantaranya adalah bentuk tidak normal, bentuk normal pertama, bentuk normal kedua dan bentuk normal ketiga.

Normalisasi digunakan untuk menentukan pengelompokkan atribut-atribut dalam sebuah relasi sehingga diperoleh relasi yang berstruktur baik. Dalam hal ini yang dimaksud dengan relasi yang berstruktur baik adalah relasi yang memenuhi dua kondisi berikut:

1) Mengandung redundansi sesedikit mungkin, dan
2) Memungkinkan baris-baris dalam relasi disisipkan,dimodifikasi dan dihapus tanpa menimbulkan kesalahan atau ketidakkonsistenan

Berikut aturan dalam dalam masing-masing tahapan *normalisasi* yang umum dan sering digunakan:

1. Bentuk Normalisasi Kesatu (1 NF/ First Normal Form)

   Bentuk Bentuk Normal Kesatu mempunyai ciri yaitu setiap data dibentuk dalam *file flat*, data dibentuk dalam satu *record* demi satu *record* dan nilai dari *field* berupa "*atomic value*". Tidak ada set atribut yang berulang ulang atau atribut bernilai ganda (*multi value*). Tiap *field* hanya satu pengertian, bukan merupakan kumpulan data yang mempunyai arti mendua. Hanya satu arti saja dan juga bukanlah pecahan kata kata sehingga artinya lain.

2. Bentuk Normal Kedua (2NF)

   Bentuk Normal kedua mempunyai syarat yaitu bentuk data telah memenuhi kriteria bentuk Normal Kesatu. Attribut bukan kunci haruslah bergantung secara fungsi pada kunci utama, sehingga untuk membentuk Normal Kedua haruslah sudah ditentukan kunci-kunci *field*. Kunci *field* harus unik dan dapat mewakili atribut lain yang menjadi anggotanya.

3. Bentuk Normal Ketiga (3NF)

   Untuk menjadi bentuk Normal Ketiga maka relasi haruslah dalam bentuk Normal Kedua dan semua atribut bukan *primer* tidak punya hubungan

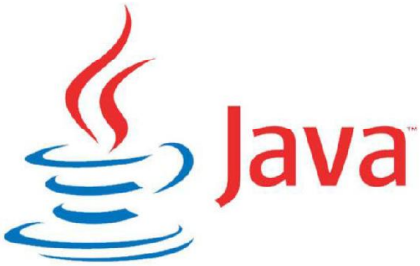
yang *transitif*. Artinya setiap atribut bukan kunci harus bergantung hanya pada kunci *primer* secara menyeluruh.

4. Boyce–Codd *Normal* Form (BCNF)

   Boyce-Codd Normal *Form* mempunyai paksaan yang lebih kuat dari bentuk Normal ketiga. Untuk menjadi BNCF, relasi harus dalam bentuk Normal Kesatu dan setiap atribut dipaksa bergantung pada fungsi pada atribut *super key*.

2.9 Tekhnologi Pengembangan Aplikasi

2.9.1 Bahasa Pemograman *Java*



Gambar 2 1 Logo *Java*

Menurut definisi *Sun Microsystem*, di dalam buku M. Shalahuddin dan Rosa A.S. (2010 : 1) *Java adalah nama sekumpulan teknologi untuk membuat dan menjalankan perangkat lunak pada komputer yang berdiri sendiri (standalone) ataupun pada lingkungan jaringan*.

Java berdiri di atas sebuah mesin penterjemah (*interpreter*) yang diberi nama *Java Virtual Machine* (JVM). JVM inilah yang akan membaca kode bit (*bytecode*) dalam file *.class* dari suatu program sebagai representasi langsung program yang berisi bahasa mesin. Oleh karena itu bahasa *Java* disebut sebagai

bahasa pemrograman yang *portable* karena dapat dijalankan pada berbagai sistem operasi, asalkan pada system operasi tersebut terdapat JVM. Alasan utama pembentukan bahasa *Java* adalah untuk membuat aplikasi-aplikasi yang dapat diletakkan di berbagai macam perangkat elektronik, sehingga *Java* harus bersifat tidak bergantung pada platform (*platform independent*). Itulah yang menyebabkan dalam dunia pemrograman *Java* dikenal adanya istilah "*write once, run everywhere*", yang berarti kode program hanya ditulis sekali, namun dapat dijalankan di bawah kumpulan pustaka (*platform*) manapun, tanpa harus melakukan perubahan kode program.

2.9.2 MySQL

Alam (2005:1) menjelaskan, *MySQL* merupakan salah program untuk mengelola *database* dalam jaringan yang sangat popular. Kunci sukses *MySQL* adalah disediakannya pilihan dua versi, yaitu versi *free software* alias gratis, dan versi *commercial license* alias dengan biaya.

Sebagai *software database* dengan konsep *database* modern, *MySQL* memiliki banyak kelebihan.

1. *Protability*

MySQL dapat digunakan dengan stabil tanpa kendala, berarti pada berbagai sistem operasi diantaranya seperti *Windows*, *Linux*, *Mac OS X Server*, *Solaris*, *Amiga HP-UX* dan masih banyak lagi.

2. *Multiuser*

MySQL dapat digunakan untuk menangani beberapa *user* dalam waktu yang bersamaan tanpa mengalami masalah atau konflik. Hal ini akan memungkinkan sebuah *database server MySQL* dapat diakses *client* secara bersamaan dalam waktu yang bersamaan pula.

3. *Performance Tuning*

MySQL memiliki kecepatan yang cukup menakjubkan dalam menangani *query* sederhana, serta mampu memproses lebih banyak *SQL* persatuan waktu.

4. *Column Types*

MySQL didukung tipe kolom(tipe data) yang sangat kompleks.

5. *Command* dan *Functions*

MySQL memiliki operator dan fungsi secara penuh yang mendukung per*int*ah *SELECT* dan *WHERE* dalam *query*.

2.9.3 *Unified Modelling Language* (UML)

Menurut (Whitten & Bentley, 2007, 371), unified modeling language (UML) merupakan kumpulan dari model yang akan digunakan untuk menjelaskan sistem dari perangkat lunak sebagai objek didalam aplikasi. Model UML yang dipakai dalam pengembangan aplikasi ini antara lain *Use Case Diagram dan Activity Diagram.*

1. *Use Case Diagram*

   Use Case Diagram mendeskripsikan sebuah *int*eraksi antara satu atau lebih aktor dengan sistem yang dibuat. Dapat dikatakan *Use Case* digunakan

untuk mengetahui fungsi yang ada di dalam sistem dan siapa saja yang berhak menggunakan fungsi-fungsi yang dibutuhkan

2. *Activity Diagram*

   Activity Diagram adalah menggambarkan berbagai alur aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masung alur berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir. *Activity diagram* juga dapat menggambarkan proses *parallel* yang mungkin terjadi beberapa eksekusi, *activity diagram* merupakan *state diagram* khusus, dimana sebagian besar *state* adalah *action.*Oleh karna itu *activity diagram* tidak menggambarkan *behavior int*ernal dalam sebuah sistem (dan *int*eraksi antara *subsistem*) secara eksak, tetapi lebih menggamarkan proses-proses dan jalur-jalur aktivitas dari level secara umum.

2.10 Perangkat Pendukung

2.10.1 *NetBeans*

Nishom (2012), menjelaskan, *Netbeans* merupakan sebuah aplikasi *Integrated Development Environment* (IDE) yang berbasiskan *Java* dari *Sun Microsystems* yang berjalan di atas *swing*. *Swing* merupakan sebuah teknologi *Java* untuk pengembangan aplikasi *desktop* yang dapat berjalan pada berbagai macam *platform* seperti *windows, linux, Mac OS X* dan *Solaris*. Sebuah IDE merupakan lingkup pemrograman yang di *int*egrasikan ke dalam suatu aplikasi perangkat lunak yang menyediakan *Graphic User Interface* (GUI), suatu kode *editor* atau *text*, suatu *compiler* dan suatu *debugger*.

Netbeans juga digunakan oleh sang *programmer* untuk menulis, *mengcompile*,mencari kesalahan dan menyebarkan program *netbeans* yang ditulis dalam bahasa pemrograman *Java* namun selain itu dapat juga mendukung bahasa pemrograman lainnya dan program ini pun bebas untuk digunakan dan untuk membuat *professional desktop, enterprise, web, and mobile applications* dengan *Java language, C/C++,* dan bahkan *dynamic languages* seperti PHP, *JavaScript, Groovy,* dan *Ruby*. NetBeans merupakan sebuah proyek kode terbuka yang sukses dengan pengguna yang sangat luas, komunitas yang terus tumbuh, dan memiliki hampir 100 mitra (dan terus bertambah!). *Sun Microsystems* mendirikan proyek kode terbuka *NetBeans* pada bulan Juni 2000 dan terus menjadi sponsor utama. Dan saat ini pun *Netbeans* memiliki 2 produk yaitu *Platform Netbeans* dan *Netbeans* IDE. Platform Netbeans merupakan framework yang dapat digunakan kembali (*reusable*) untuk menyederhanakan pengembangan aplikasi *desktop* dan Platform NetBeans juga menawarkan layanan-layanan yang umum bagi aplikasi *desktop*, mengijinkan pengembang untuk fokus ke logika yang spesifik terhadap aplikasi.

2.10.2 XAMPP

Xampp merupakan suatu *software* yang di dalamnya terdapat *Apache* yang berfungsi sebagai *web server*, *PHP* (*Hypertext Preprocessor*) merupakan bahasa *web server s*id*e* yang bersifat *open source* dan *MySQL* adalah basis data yang menghubungkan script *PHP* menggunakan per*int*ah *query* dan *escape character* yang sama dengan *PHP*. *PHP* memang mendukung banyak jenis *basis data*, tetapi untuk membuat sebuah *basis data* yang dinamis dan selalu *up to date*, *MySQL*

merupakan pilihan *basis data* tercepat saat ini. Selain itu terdapat juga *PhpMyAdmin* sebagai tempat melakukan konfigurasi keseluruhan. (Sidik, 2006).