

BAB II

LANDASAN TEORI

Berikut ini akan dijelaskan beberapa pengertian dan teori yang berhubungan dengan laporan skripsi ini antara lain pengertian dasar yang berkaitan dengan judul, teori umum pengembangan aplikasi dan sekilas teori tentang bahasa.

2.1 Sistem Informasi Pembayaran Pajak

1. Rancang Bangun

Rancang Bangun adalah proses mengatur, melakukan atau merencanakan segala sesuatu sebelum bertindak untuk memecahkan permasalahan yang ada. (Kamus Bahasa Indonesia, 2005).

2. Sistem informasi

Sistem informasi adalah suatu sistem di dalam suatu organisasi yang mempertemukan kebutuhan pengolahan transaksi harian yang mendukung fungsi organisasi yang bersifat manajerial dalam kegiatan strategi dari suatu organisasi untuk dapat menyediakan kepada pihak luar tertentu dengan laporan – laporan yang diperlukan. (Jogiyanto, 2005)

3. Pajak

Pajak berdasarkan SE-19/PJ/2007 Sistem Informasi Direktorat Jenderal Pajak adalah “suatu sistem informasi administrasi perpajakan di lingkungan kantor Direktorat Jenderal Pajak modern dengan menggunakan perangkat keras dan perangkat lunak yang dihubungkan dengan suatu jaringan kerja di kantor pusat”. Pengertian pajak menurut Rochmat Soemitro dalam buku Perpajakan Edisi Revisi (2009) “Pajak adalah iuran rakyat kepada kas Negara berdasarkan undang-undang (yang dapat dipaksakan) dengan tiada mendapat jasa timbal (kontraprestasi) yang langsung dapat diajukan dan yang digunakan untuk membayar pengeluaran umum ”Sementara menurut Supramono dan Theresia Woro Damayanti (2010) dalam bukunya Perpajakan Indonesia Mekanisme dan Perhitungan, menjelaskan bahwa: “Pajak merupakan iuran dari rakyat kepada negara”.

4. Surat Ijin Mengemudi

Surat Ijin Mengemudi yang selanjutnya disingkat SIM adalah tanda bukti legitimasi kompetensi, alat kontrol dan data forensik Kepolisian bagi seseorang yang telah lulus uji pengetahuan, kemampuan dan keterampilan untuk mengemudikan Ranmor di jalan sesuai dengan persyaratan yang ditentukan berdasarkan Undang-Undang Lalu Lintas dan Angkutan Jalan; (<http://ferli1982wordpress.com>).

5. Mutasi Kendaraan

Pengertian MUTASI KENDARAAN BERMOTOR adalah perpindahan administrasi identifikasi kendaraan bermotor dari suatu daerah ke daerah lain sesuai dengan perpindahan alamat baru pemilik kendaraan bermotor. (<http://kioshukumonline.blogspot.com>).

2.2 Metodologi Pengembangan Sistem

Metodologi pengembangan sistem adalah suatu cara atau metode yang digunakan untuk melakukan suatu hal pendekatan sistem merupakan metodologi dasar untuk memecahkan masalah. (Jogiyanto, 2005)

Pengembangan sistem (*system development*) dapat berarti menyusun suatu sistem yang baru untuk menggantikan sistem yang lama secara keseluruhan atau memperbaiki sistem yang telah ada. Berikut ini adalah alasan mengapa suatu sistem perlu diperbaiki atau diganti.

a. Adanya permasalahan-permasalahan (*problem*) yang timbul di sistem yang lama. Permasalahan yang timbul dapat berupa:

1. Ketidakberesan

Ketidakberesan pada sistem yang lama tidak dapat beroperasi sesuai yang diharapkan. Ketidakberesan ini dapat berupa kesalahan-kesalahan yang tidak disengaja yang juga dapat menyebabkan tidak validnya data yang diperlukan.

2. Pertumbuhan Organisasi

Pertumbuhan organisasi yang menyebabkan harus disusunnya sistem yang baru. Pertumbuhan organisasi diantaranya adalah kebutuhan informasi yang semakin luas, volume pengolahan data semakin meningkat, perubahan prinsip akuntansi yang baru.

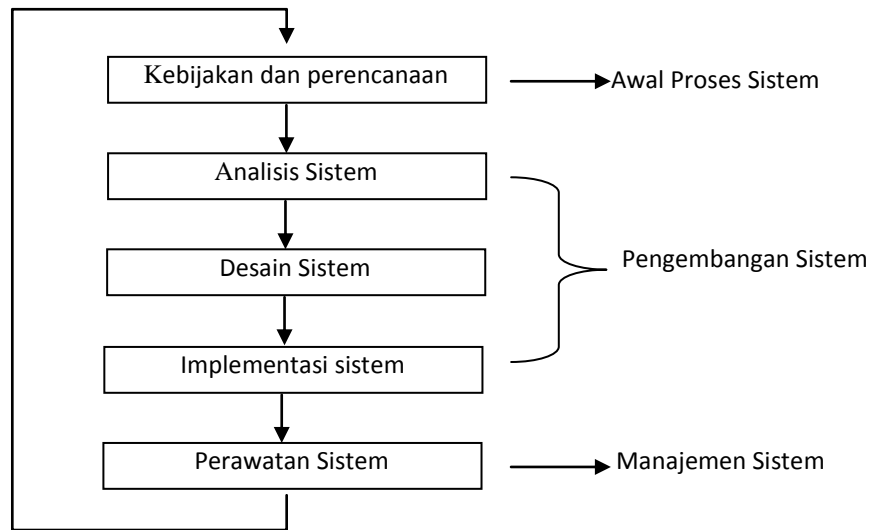
Karena adanya perubahan ini, maka menyebabkan sistem lama menjadi tidak dapat lagi memenuhi kebutuhan informasi yang dibutuhkan manajemen.

- b. Adanya instruksi-instruksi yaitu penyusunan sistem yang baru dapat juga terjadi karena adanya instruksi-instruksi dari pimpinan atau dari luar organisasi, misalnya peraturan pemerintah.

2.2.1 Model Pengembangan Sistem

Metodologi pengembangan sistem adalah metode-metode, prosedur-prosedur, konsep-konsep, pekerjaan, aturan-aturan yang akan digunakan untuk mengembangkan suatu sistem informasi. Pada penulisan skripsi ini menggunakan metodologi analisis dan desain sistem terstruktur SSAD (*Structured Systems Analysis and Design*) untuk digunakan pada pengembangan sistem. Metodologi ini dapat digunakan pada tahap analisis dan tahap desain dan metodologi ini menggunakan pendekatan pengembangan sistem terstruktur yang dilengkapi dengan alat-alat (*tools*) dan teknik-teknik (*techniques*) yang dibutuhkan dalam pengembangan sistem, sehingga hasil akhir dari sistem yang dikembangkan akan didapatkan sistem yang strukturnya didefinisikan dengan baik dan jelas. Metodologi ini secara umum didasarkan pada pemecahan dari sistem ke dalam modul-modul berdasarkan dari tipe elemen data dan tingkah laku logika modul tersebut di dalam sistem. Dengan metodologi ini, sistem secara logika dapat digambarkan secara logika dari arus data dan hubungan antar fungsinya di dalam modul-modul sistem (Jogiyanto, 2005). Pengembangan sistem informasi yang berbasis komputer dapat merupakan tugas kompleks yang membutuhkan banyak sumber daya dan dapat memakan waktu berbulan-bulan bahkan bertahun-tahun untuk menyelesaikannya. Proses pengembangan sistem melewati beberapa tahapan dari mulai sistem itu direncanakan sampai dengan sistem tersebut

diterapkan, dioperasikan dan dipelihara. Siklus ini disebut dengan siklus hidup pengembangan sistem (Jogiyanto, 2005). Siklus hidup pengembangan sistem dengan langkah-langkah utamanya dapat dilihat pada gambar 2.1.



Gambar 2.1 Siklus Hidup Pengembangan Sistem

Berikut tahap-tahap Penjelasan siklus hidup pengembangan system :

(a). Kebijakan dan Perencanaan

Pada tahap ini bertujuan untuk mengidentifikasi dan memprioritaskan sistem informasi apa yang akan dikembangkan, sasaran-sasaran yang ingin dicapai, jangka waktu pelaksanaan serta mempertimbangkan dana yang tersedia dan siapa yang melaksanakan.

(b). Analisis Sistem

Pada saat perencanaan telah selesai dan mekanisme pengendalian telah berjalan, tim proyek beralih pada analisis sistem yang telah ada. Analisis sistem adalah penelitian atas sistem yang telah ada dengan tujuan untuk merancang sistem yang baru atau diperbarui.

(c). Desain Sistem

Desain sistem ini meliputi penentuan pemrosesan dan data yang dibutuhkan oleh sistem yang baru, dan pemilihan konfigurasi terbaik dari hardware yang menyediakan desain. Desain system adalah ketentuan mengenai proses dan data yang dibutuhkan oleh sistem yang baru.

(d). Implementasi Sistem

Pada tahap ini melibatkan beberapa spesialis informasi tambahan yang mengubah desain dari bentuk kertas menjadi satu dalam hardware, software, dan data. Implementasi adalah penambahan dan penggabungan antara sumber-sumber secara fisik dan konseptual yang menghasilkan pekerjaan sistem.

(e). Perawatan Sistem

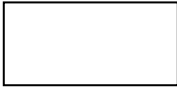


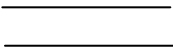
Tahap terakhir perawatan system ini bertujuan untuk memastikan bahwa sistem yang digunakan oleh pihak pengguna benar-benar telah stabil dan terbebas dari *error* dan *bug*. Pemeliharaan ini biasanya berkaitan dengan masa garansi yang diberikan oleh pihak pengembangan sistem sesuai dengan perjanjian dengan pihak pengguna.

2.2.2 Alat-Alat Dalam Pengembangan Sistem

Pada sub bab ini akan diuraikan mengenai alat-alat yang digunakan dalam pengembangan sistem antara lain *data flow diagram*, bagan alir dokumen, dan kamus data.

1. DFD (*Data Flow Diagram*)

Data Flow Diagram (DFD) adalah gambaran grafis yang memperlihatkan aliran data dari sumbernya dalam obyek kemudian melewati suatu proses yang mentransformasikan ke tujuan yang lain, yang ada pada objek lain. Simbol yang digunakan dalam DFD (*Data Flow Diagram*) dapat dilihat pada Tabel 2.1.

Simbol	Keterangan
(external entity) 	Merupakan sumber atau tujuan dari aliran data dari atau ke sistem
Arus data (data flow) 	Menggambarkan aliran data dari satu proses ke proses lainnya
Proses (process) 	Proses atau fungsi yang menstransformasikan data
Simpanan data (data store) 	Komponen yang berfungsi untuk menyimpan data atau file.

Tabel 2.1 Simbol DFD (*Data Flow Diagram*)

Penjelasan :

a. Kesatuan Luar (*External Entity*)

Setiap sistem mempunyai batasan sistem (*boundary*) yang memisahkan suatu sistem dengan lingkungan luarnya. Kesatuan luar (*external entity*) merupakan kesatuan (*entity*) di lingkungan luar sistem yang dapat berupa orang, organisasi atau sistem lainnya yang berada di lingkungan luarnya yang akan memberikan *input* atau menerima *output* dari sistem.

b. Arus Data (*Data flow*)

Arus data (*data flow*) di DFD diberi simbol suatu panah. Arus data ini mengalir diantara proses, simpanan data, dan kesatuan luar.

c. Proses (*Process*)


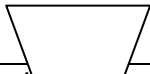
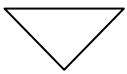


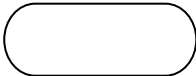
Suatu proses adalah kegiatan atau kerja yang dilakukan oleh orang, mesin atau komputer dari hasil suatu arus data yang masuk ke dalam proses untuk dihasilkan arus data yang akan keluar dari proses.

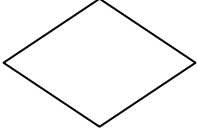

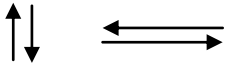
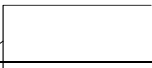

d. Simpanan Data (*Data Store*)

Simpanan data (*data store*) adalah simpanan suatu data yang dapat berupa *file*, arsip.

2. Bagan Alir Dokumen

Bagan alir dokumen (*Document flowchart*) merupakan bagan alir yang menunjukkan arus data dari laporan dan formulir termasuk tembusan-tembusannya. (Jogiyanto, 2005) Bagan alir dokumen juga menggambarkan bagian-bagian yang terkait dalam proses yang sedang berlangsung atau yang sedang berjalan, proses apa yang menghasilkan proses tersebut dan interaksi antara data yang tersimpan dan proses yang dikenakan pada data tersebut. Simbol – simbol bagan alir dokumen yang digunakan antara lain sebagai berikut

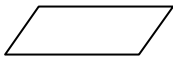
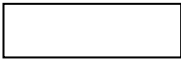
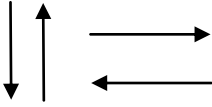
Simbol	Keterangan
Simbol dokumen 	Menandakan dokumen, bisa dalam bentuk surat, formulir, buku/bendel/berkas atau cetakan
Simbol kegiatan manual 	Proses manual.
Simbol simpanan / arsip 	Menunjukkan dokumen yang diarsipkan (arsip manual)
Simbol proses 	Proses yang dilakukan oleh komputer
Simbol Data Storage 	Data penyimpanan (<i>Data Storage</i>)
Simbol terminasi 	Terminasi yang menandakan awal dan akhir dari suatu aliran


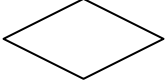



Simbol keputusan 	Pengambilan keputusan (<i>Decision</i>)
Simbol pemasukan 	Pemasukan data secara manual
Simbol garis alir 	Menunjukkan arus dari data
Simbol penjelasan 	Menunjukkan penjelasan dari suatu proses.
Simbol penghubung 	Menunjukkan penghubung kehalaman yang masih sama atau ke halaman lain.

Tabel 2.2 Bagan Alir Dokumen

3. Bagan Alir Program (*Program Flowchart*)

Bagan alir program (program flowchart) adalah bagian flowchart yang menggambarkan arus logika dari data yang akan diproses kedalam suatu program mulai dari awal sampai akhir. Bagan alir merupakan alat yang berguna bagi programmer untuk mempersiapkan program yang rumit. (Jogiyanto, 2005). Simbol – simbol bagan alir program yang digunakan antara lain sebagai berikut:

Simbol	Keterangan
Input / Output 	Simbol <i>input/output</i> digunakan untuk mewakili data <i>input/output</i>
Proses 	Simbol proses digunakan untuk mewakili suatu proses.
Garis Alir 	Simbol garis alir (<i>flow lines symbol</i>) digunakan untuk menunjukkan arus dari proses

Penghubung 	Simbol penghubung (<i>connector symbol</i>) digunakan untuk menunjukkan sambungan dari bagan alir yang terputus dihalaman yang sama / dihalaman yang lain
Keputusan 	Simbol keputusan (<i>decision symbol</i>) digunakan untuk suatu penyelesaian kondisi di dalam program
Proses terdefinisi 	Simbol proses terdefinisi digunakan untuk menunjukkan suatu operasi yang rinciannya ditunjukkan di tempat lain
Persiapan 	Simbol persiapan (<i>preparation symbol</i>) digunakan untuk memberi nilai awal suatu besaran.
Terminal 	Simbol terminal (<i>terminal symbol</i>) digunakan untuk menunjukkan awal dan akhir dari suatu proses / program

Tabel 2.3 Bagan alir program (*Program Flowchart*)

4. Kamus Data

Kamus data atau data *dictionary* adalah catalog fakta tentang data dan kebutuhan informasi dari suatu sistem informasi. Selama penyusunan suatu system informasi, kamus data digunakan sebagai alat untuk mendefinisikan aliran data yang mengalir di sistem, merancang input, merancang laporan-laporan dan merancang database. (Jogyanto, 2005)

Pembentukan kamus data didasarkan pada alur data yang terdapat pada DFD. Alur data pada DFD bersifat global (hanya menunjukkan nama alur datanya tanpa menunjukkan struktur dari alur data). Untuk menunjukkan struktur dari alur data secara rinci maka dibentuklah kamus data.

Format DD

Nama Database :
 Nama Tabel :
 Primary Key :
 Foreign Key :

Nama Field	Type	Size	Kondisi	Keterangan

Keterangan: Kondisi berisi (contoh: NULL/NOT NULL)

Gambar 2.2 Format Kamus Data

2.3 Basis Data

Basis data adalah sekumpulan subsistem yang terdiri atas basis data secara bersama - sama, personal-personal yang merancang dan mengelola basis data, teknik-teknik untuk merancang dan mengelola basis data, serta sistem komputer untuk mendukungnya. Jadi, basis data adalah kumpulan file yang saling berkaitan yang disimpan secara bersama-sama pada suatu media dan memiliki kunci dari tiap filenya. Dilihat dari jenisnya, basis data dibagi menjadi dua yaitu: (Waljiyanto, 2004)

1. Basis data flat-file.

Basis data flat-file ideal untuk data berukuran kecil dan dapat dirubah dengan mudah. Pada dasarnya, mereka tersusun dari sekumpulan string dalam satu atau lebih file yang dapat diurai untuk mendapatkan informasi yang disimpan. Basis data flat-file baik digunakan untuk menyimpan daftar atau data yang sederhana dan dalam jumlah kecil. Basis data flat-file akan menjadi sangat rumit apabila digunakan untuk menyimpan data dengan struktur kompleks walaupun dimungkinkan pula untuk menyimpan data semacam itu. Salah satu masalah menggunakan basis data jenis ini adalah rentan pada korupsi data karena tidak adanya penguncian yang melekat ketika data digunakan atau dimodifikasi.

2. Basis data relasional.

Basis data ini mempunyai struktur yang lebih logis terkait cara penyimpanan. Kata "relasional" berasal dari kenyataan bahwa tabel-tabel yang berada di basis data dapat dihubungkan satu dengan lainnya. Basis data relasional menggunakan sekumpulan tabel dua dimensi yang masing-masing tabel tersusun atas baris (tupel) dan kolom (atribut). Untuk membuat hubungan antara dua atau lebih tabel, digunakan key (atribut kunci) yaitu *primary key* di salah satu tabel dan *foreign key* di tabel yang lain. Saat ini, basis data relasional menjadi pilihan karena keunggulannya. Beberapa kelemahan yang mungkin dirasakan untuk basis data jenis ini adalah implementasi yang lebih sulit untuk data dalam jumlah besar dengan tingkat kompleksitasnya yang tinggi dan proses pencarian informasi yang lebih lambat karena perlu menghubungkan tabel-tabel terlebih dahulu apabila datanya tersebar di beberapa tabel.

Tori-teori yang perlu diperhatikan untuk merancang database adalah:

1. *Entitas* : suatu objek yang dapat dibedakan dengan yang lainnya;
2. *Atribut* : karakteristik yang menjadi ciri *entitas*;
3. *Field* : suatu unit informasi mengenai suatu *entitas* yang mempunyai arti;
4. *Record* : kumpulan dari suatu *field* informasi mengenai *entitas* tertentu atau kumpulan dari item data yang saling berhubungan;
5. *File* : kumpulan *record* yang saling berhubungan;

Penyusunan basis data digunakan untuk mengatasi masalah-masalah pada penyusunan data antara lain sebagai berikut:

1. Redudansi dan Inkonsistensi data

Yaitu jika *file-file* dan program aplikasi diciptakan oleh *programer* yang berbeda pada waktu yang berselang cukup panjang, maka ada beberapa bagian data yang mengalami penggandaan (*redudancy*) pada *file* yang berbeda. Penyimpanan data yang sama berulang-ulang di beberapa *file* dapat mengakibatkan juga inkonsistensi atau tidak konsisten, hal ini terjadi satu *file record* diubah tanpa mengubah *file* yang sama pada *record* yang lain.

2. Kesulitan Pengaksesan Data

Kesulitan pengaksesan data timbul bila suatu saat terjadi pengolahan data yang kompleks dan dalam jumlah yang besar, sementara belum tersedia program untuk menunjang hal itu.

3. Isolasi Data Standarisasi

Jika data terbesar beberapa *file* dalam *format* yang tidak sama, maka ini akan menyulitkan dalam menulis program aplikasi untuk mengambil dan menyimpan data, maka data harus dalam satu *database* yang dibuat satu *format* sehingga mudah dibuat program aplikasi.

4. Banyak Pemakaian (*MultipleUser*)

Dalam rangka mempercepat daya guna sistem dan mendapat respon waktu cepat, beberapa sistem mengizinkan banyak pemakaian untuk meng-*update* data nantinya, akan dipakai dalam waktu yang berbeda.

5. Masalah Kesatuan (*integritas*)

Database berisi *file-file* yang saling berkaitan, masalah utamanya adalah bagaimana kaitan antara *file* tersebut terjadi.


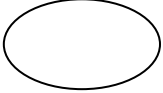
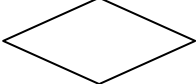

6. Masalah Keamanan (*Security*)

Setiap merancang suatu *database*, masalah keamanan atau *security* harus sangat diperhatikan agar setiap pemakai (*User*) tidak dapat mengakses semua data. (Kristanto,2003).

Berikut ini adalah teori-teori basis data yang digunakan pada penelitian ini:

a. *Entity Relationship*

Entity Relationship merupakan suatu model data yang dikembangkan berdasarkan obyek. *Entity relationship* digunakan untuk menjelaskan hubungan antar data dalam basis data kepada pemakai secara logik. Simbol yang digunakan antara lain :

Simbol	Keterangan
Entity 	Menyatakan himpunan entitas.
Atribut 	Menyatakan atribut yang berfungsi sebagai <i>key</i> .
Relasi 	Menyatakan himpunan relasi.
Penghubung 	Menyatakan penghubung antara himpunan relasi dengan himpunan entitas dan himpunan entitas dengan atributnya.

Tabel 2.3.1 Simbol Entity Relationship

Hubungan yang terjadi pada suatu tabel dengan tabel yang lainnya, yang berfungsi untuk mengatur operasi suatu database. Hubungan yang dapat dibentuk dapat mencakupi 3 (tiga) macam hubungan yaitu :

1. One-To-One (1 – 1)

Mempunyai pengertian “Setiap baris data pada tabel pertama dihubungkan hanya ke satu baris data pada tabel ke dua”.

2. One-To-Many (1 –)

Mempunyai pengertian “Setiap baris data dari tabel pertama dapat dihubungkan ke satu baris atau lebih data pada tabel ke dua”.

3. Many-To-Many (–)

Mempunyai pengertian “Satu baris atau lebih data pada tabel pertama bisa dihubungkan ke satu atau lebih baris data pada tabel ke dua”.

b. Normalisasi

Proses normalisasi adalah suatu proses dimana elemen-elemen data dikelompokkan menjadi tabel-tabel, dimana dalam tabel tersebut terdapat entity-entity dan relasi antar entity tersebut (Husni Iskandar, 2004).

Langkah-langkah dalam perancangan struktur basis data dengan menggunakan teknik normalisasi adalah sebagai berikut.

1. Membuat bentuk un-normal (*un-normalized form*)

Yaitu dengan cara memasukan seluruh atribut yang diperlukan kedalam satu *file* atau relasi kemudian tentukan atribut kuncinya (*key field*).

2. Membuat bentuk normal pertama (*1NF*)

Bentuk normal pertama adalah suatu relasi yang tidak mengandung grup berulang (*repeating group*). Untuk mendapatkan normal pertama adalah dengan cara memisahkan grup berulang ke dalam relasi baru, kemudian kunci utamanya (*primary key*) adalah kunci utama grup berulang ditambah kunci utama relasi asal.

3. Membuat bentuk normal kedua (*2NF*)

Sebuah relasi dikatakan dalam bentuk normal kedua bila relasi tersebut dalam bentuk normal pertama serta seluruh atribut (yang bukan kunci utama) tergantung secara fungsional sepenuhnya pada kunci utama (tidak hanya pada sebagian kunci utama). Untuk mendapatkan bentuk normal kedua adalah dengan cara memisahkan atribut yang tergantung secara fungsional pada sebagian kunci utama kedalam relasi baru, kemudian tambahkan atribut yang mengidentifikasinya.

4. Membuat bentuk normal ketiga (*3NF*)

Suatu relasi dikatakan dalam bentuk normal ketiga jika relasi tersebut dalam bentuk normal kedua dan setiap atributnya tidak tergantung secara transitif pada kunci utama. Untuk mendapatkan bentuk normal ketiga adalah dengan cara memisahkan atribut yang mempunyai ketergantungan transitifitas ke dalam relasi baru, kemudian tambahkan atribut yang mengidentifikasinya.

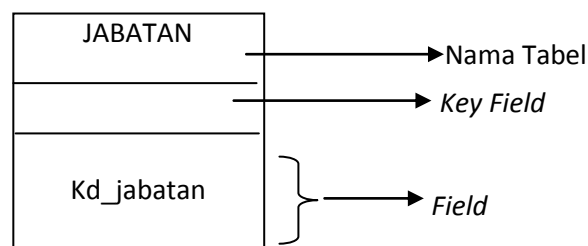
5. Membuat bentuk normal keempat (*4NF*)

Suatu relasi dikatakan dalam bentuk normal keempat jika relasi tersebut dalam bentuk normal ketiga dan seluruh atribut (yang bukan kunci utama) tidak tergantung bernilai banyak pada kunci utama (*multi valued dependencies*). Untuk mendapatkan normal keempat adalah dengan cara memisahkan atribut yang mempunyai ketergantungan nilai banyak ditambah kunci utama relasi asal menjadi kunci utama relasi baru.

2.4 Relasi Antar Tabel

Menurut Jogiyanto (2005) Suatu file yang terdiri dari beberapa grup elemen yang berulang-ulang perlu diorganisasikan kembali. Proses untuk mengorganisasikan file untuk menghilangkan grup elemen yang berulang-ulang dan digambarkan dalam bentuk database struktur hubungan disebut Relasi Antar Tabel. Database dengan struktur data hubungan dapat digambarkan dalam bentuk dua dimensi. Kolom dari tabel menunjukkan atribut dari file. Atribut ini menunjukkan item

data atau *field*. Kumpulan nilai dari *field* atau item data disebut dengan istilah *domain*. Masing-masing baris dari record di dalam tabel disebut dengan istilah *tuple*. Suatu *tuple (record)* yang mempunyai dua domain disebut dengan 2 tuple. Suatu tuple yang mempunyai 3 domain disebut dengan 3-tuple dan seterusnya. Tiap-tiap tuple atau record ini dapat mempunyai suatu kunci yang unik dengan cara mana tuple ini dapat diidentifikasi. Field yang menjadi kunci yang unik ini disebut dengan field kunci (*key field*).



Contoh Relasi Antar Tabel.

2.5 Sistem Pengkodean

Sistem pengkodean digunakan tujuan mengklasifikasikan data, memasukkan data ke dalam komputer dan untuk mengambil bermacam-macam informasi yang berhubungan dengannya. Kode dapat dibentuk dari kumpulan angka, huruf dan karakter-karakter khusus. Angka merupakan simbol yang banyak digunakan pada sistem pengkodean. Aka tetapi kode yang berbentuk angka lebih dari 6 digit akan sangat sulit untuk diingat. Kode numerik menggunakan 10 macam kombinasi angka di dalam kode. Kode alfabetik menggunakan 26 kombinasi huruf untuk koenya. Kode alphanumerik merupakan kode yang meggunakan gabungan angka, huruf dan karakter khusus. (Jogiyanto, 2005).

2.5.1 Petunjuk Pembuatan Kode

Di dalam merancang suatu kode harus diperhatikan beberapa hal, yaitu sebagai berikut :

1. Harus mudah diingat
Supaya kode mudah diingat, maka dapat dilakukan dengan cara menghubungkan kode tersebut dengan obyek yang diwakili dengan kodenya.
2. Harus unik
Kode harus unik untuk masing-masing item yang diwakilinya. Unik berarti tidak ada kode yang kembar.
3. Harus fleksibel
Kode harus fleksibel sehingga memungkinkan perubahan atau penambahan item baru dapat tetap diwakili oleh kode.
4. Harus efisien
Kde harus sependek mungkin, selain mudah diingat juga akan efisien bila direkam di simpanan luar komputer.
5. Harus konsisten
Bilamana mungkin, kode harus konsisten dengan kode yang telah dipergunakan pemasok saja, maka dapat dipergunakan kode-kode barang ang sudah dipergunakan oleh pemasok.
6. Harus distandarisasi
Kode harus distandarisasi untuk seluruh tingkatan departemen dalam organisasi.
7. Spasi harus dihindari
Spasi di dalam kode sebaiknya dihindari, karnea dapat menyebabkan kesalahan di dalam meggunakannya.
8. Hindari karakter yang mirip
Karakter yang hampir serupa bentuk dan bunyi pengucapannya sebaiknya tidak digunakan dalam kode.
9. Panjang kode harus sama

Masing-masing kode yang sejenis harus mempunyai panjang yang sama.

2.5.2 Tipe dari Kode

Ada beberapa macam tipe dari kode yang dapat digunakan di dalam sistem informasi, di antaranya adalah sebagai berikut :

a. Kode Memonik

Kode mnemonik (*mnemonic code*) digunakan untuk tujuan supaya mudah diingat. Kode mnemonik dibuat dengan dasar singkatan atau mengambil sebagian karakter dari item yang akan diwakili dengan kode ini. Misalnya kode "P" untuk Pelabuhan.

b. Kode Urut

Kode urut (*sequential code*) disebut juga dengan kode seri (*serial code*) merupakan kode yang nilainya urut antara satu kode dengan kode berikutnya :

Contoh :001 : id Anggota

c. Kode Blok

Kode blok (*block code*) mengklasifikasikan item ke dalam kelompok blok tertentu yang mencerminkan satu klasifikasi tertentu atas dasar pemakaian maksimum yang diharapkan.

d. Kode Group

Kode group (*group code*) merupakan kode yang berdasarkan field-field dan tiap-tiap field kode mempunyai arti.

e. Kode Desimal

Kode desimal (*decimal code*) mengklasifikasikan kode atas dasar 10 unit angka desimal dimulai dari angka 0 sampai dengan angka 9 atau dari 00 sampai dengan 99 tergantung dari banyaknya kelompok.

2.6 Teori Pengembangan Rancang Bangun

Pada sub ini akan menjelaskan beberapa teori mengenai pengembangan rancang bangun yang akan mendukung sistem informasi, seperti bahasa pemrograman yang akan digunakan serta aplikasi pendukung, sebagai berikut :

2.6.1 Sekilas mengenai *Microsoft Visual Basic Net 2005*

Microsoft Visual Basic NET adalah teknologi pemrograman microsoft yang dapat digunakan untuk membuat aplikasi di lingkungan kerja berbasis windows. Visual basic net 2005 adalah pengembangan dari visual basic sebelumnya. Kelebihan VB Net 2005 terletak pada tampilannya yang lebih canggih dibandingkan visual basic edisi sebelumnya. Selain memiliki kelebihan VB Net 2005 memiliki kekurangan yaitu beratnya aplikasi ini apabila dijalankan pada komputer yang memiliki spesifikasi sederhana. (Yuswanto, 2006)

2.6.2 *Microsoft SQL Server 2000*

Microsoft SQL Server adalah salah satu produk *Relational Database* yang mendukung pemakaian *SQL (Structured Query Language)* dan dirancang untuk penggunaan aplikasi dengan arsitektur *client/server*. Fungsi utamanya adalah sebagai *database server* yang mengatur semua proses penyimpanan data dan transaksi suatu aplikasi (Bambang Robi'in, 2005). Kelebihan dan kekurangan dari *Microsoft SQL Server* adalah sebagai berikut :

Kelebihan pemrograman *SQL Server 2000*

- a) Merupakan *database system* yang dirancang untuk pemrograman berbasis jaringan.
- b) Keamanan data lebih terjamin, karena *file database* yang tersimpan tidak dapat dihapus bilamana *database SQL Server 2000* masih aktif.
- c) Mampu menyimpan data lebih banyak.
- d) Memberikan *tools type* data yang banyak dan lebih rinci.

Kekurangan *Microsoft SQL Server 2000* diantaranya yaitu

- a) Hanya dapat dijalankan pada sistem operasi *windows*.
- b) Aplikasi yang dibuat tidak dapat dijalankan bila komputer tersebut belum diinstall *database SQL Server 2000*.

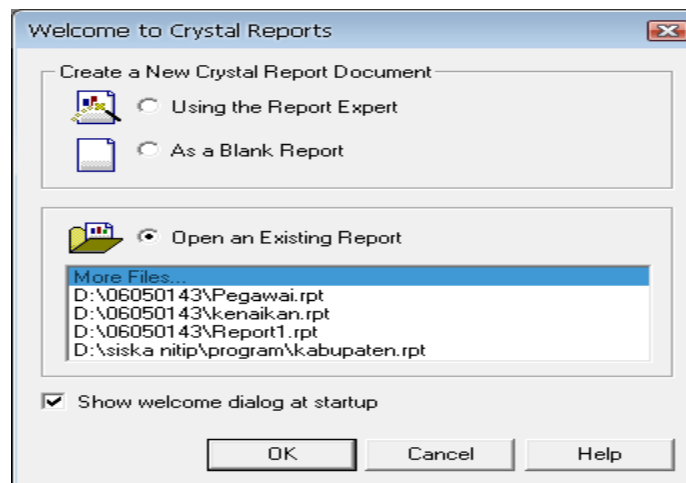
2.6.3 *Crystal Report*

Crystal report merupakan salah satu produk dari *Seagate software* yang menangani perkembangan teknologi penyajian laporan. Berikut ini adalah dasar-dasar cara pengoperasian *crystal report* :

- a) Membuat *report* baru dengan *blank report*

Membuat laporan menggunakan *blank report*. Disamping dengan cara ini, ada cara lain untuk membuat *report*, yaitu dengan modifikasi *report* yang sudah ada sebelumnya. Untuk membuat *blank report*, langkah-langkahnya adalah sebagai berikut:

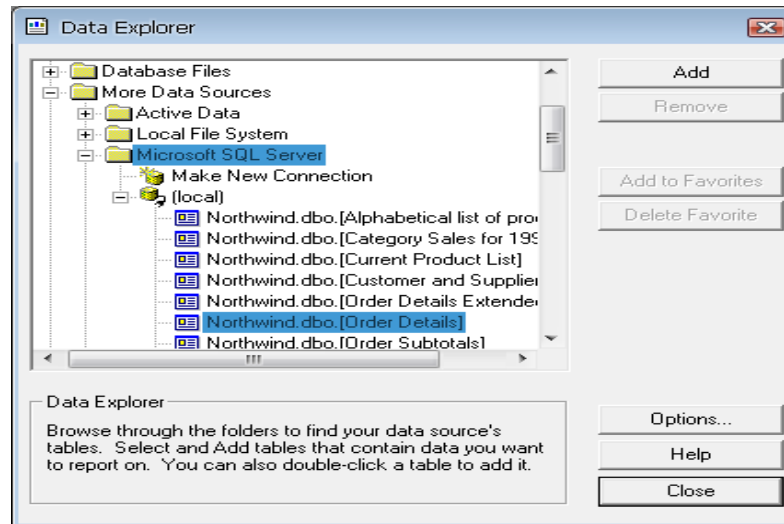
1. Buka *crystal report* dan aktifkan pilihan *As a Blank Report*. Lihat pada gambar 2.3.



Gambar 2.3 *As a Blank Report*

2. Tentukan dari mana *crystal report* akan mengambil data. Kita akan menggunakan *database SQL server* sebagai sumber data. Untuk mengoneksi kesumber data di gunakan *ODBC*.

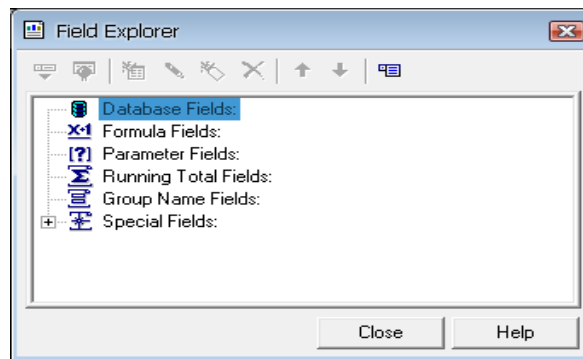
3. Scroll daftar dari *Data Sources* yang ada sampai anda menemukan *Northwind* (*database* demo yang ada pada *SQL server*). Bila tidak ada maka anda dapat menambahkannya secara manual melalui *Star + Setting + Control Panel + Data Sources* selanjutnya ikuti wizard untuk menambahkan *data source* tersebut sampai *Northwind* tampak pada *tree ODBC*. Lihat pada gambar 2.20.



Gambar 2.4 *Data Sources*

4. Sekali saja *DSN* yang benar di pilih, maka langkah berikutnya adalah memilih tabel dari *database* yang akan digunakan sebagai sumber data pembuatan *report*. Sebagai contoh anda dapat memilih tabel *order* dari daftar lalu klik tombol *Add*.
 - a. Kemudian tekan tombol *Close*.
 - b. Selanjutnya akan disajikan *windows Field Explorer*.

Lihat pada gambar 2.5.



Gambar 2.5 Windows Field Explorer

- c. Area design pada *crystal report* di bagi menjadi 5 section, yaitu: *report header*, *page header*, *details*, *report footer* dan *page footer*.
- d. Perhatikan *field explorer*. Pastikan tabel *order* dan *field-field* yang tampak. Selanjutnya pilih dengan klik dan kemudian *drag field* yang anda pilih. Sebagai contoh, klik dan *drag OrderID* ke *section details*. Lihat pada gambar 2.6.



Gambar 2.6 section details

b) Menyimpan *Report*

Untuk menyimpan *report* yang telah anda buat, lakukan dengan memilih *File* + *Save* + *Save As* dari menu utama. Simpan *report* tadi dengan nama *MyOrder.RPT* pada *hardisk* lokal anda.

c) Mengatur posisi dan ukuran *object*

Sebelum mengubah posisi dan ukuran *object*, terlebih dahulu tambahkan beberapa *field* pada *report* dengan cara sebagai berikut:

- 1) Tarik beberapa *field* tambahkan ke *section details*.
- 2) Selanjutnya tekan *priview* untuk melihat *report* yang dibuat.

