BAB 2
TINJAUAN PUSTAKA

2.1 Tinjauan Umum Perusahaan

Tinjauan umum perusahaan merupakan pembahasan mengenai sejarah, struktur organisasi, serta aspek kegiatan pada perusahaan yang akan menjadi tempat dibangunnya system ini.

2.1.1 Tentang Perusahaan

CV. Avis Car Accessories merupakan perusahaan yang sedang berkembang dan bergerak dibidang penjualan barang dengan menawarkan berbagai variasi mobil yang dapat digunakan pada berbagai jenis kendaraan,seperti kulit jok,variasi stir mobil, klakson mobil, dan berbagai variasi kendaraan lainnya.

2.1.2 Sejarah Perusahaan

CV. Avis Car Accessories berdiri sejak tahun 2008 dan terletak di Jl. Soekarno Hatta No.372 Bandung yang semula bernama Bintang Karisma dan pada tahun 2011 berganti nama menjadi Avis Car Accessories.

2.1.3 Visi Misi dan Tujuan Perusahaan

Pada sub-bab ini akan menjelaskan tentang Visi, Misi dan Tujuan dari CV. Avis Car Accessories.

2.1.3.1 Visi

Menjunjung tinggi nilai-nilai kejujuran dalam berbisnis dan mengutamakan kualitas barang, sehingga menjadikan salah satu perusahaan yang maju,produktif dan berkompetitif di Indonesia.

2.1.3.2 Misi

Misi dari CV. Avis Car Accessories adalah sebagai berikut:

1. Mengutamakan kejujuran.
2. Melayani konsumen dengan sepenuh hati.
3. Menjadi perusahaan terdepan pada bidangnya.
4. Memberi kualitas barang dengan harga yang pantas.

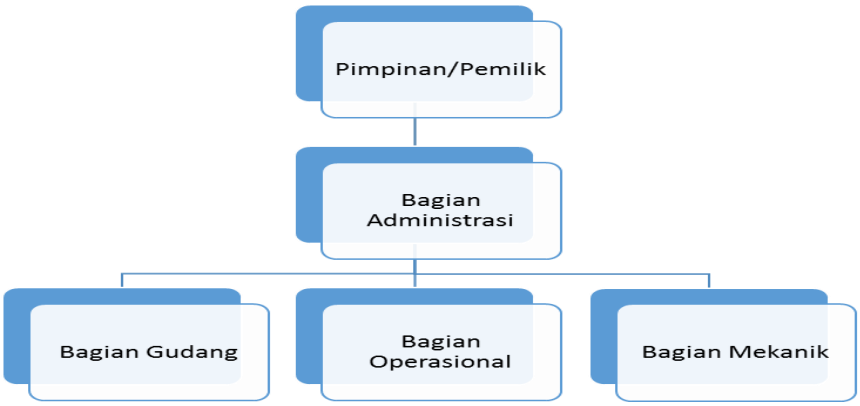
2.1.3.3 Tujuan Perusahaan

Tujuan dari CV. Avis Car Accessories adalah sebagai berikut:

1. Memproduksi dan memasarkan semua produk untuk mendapatkan laba yang optimal.
2. Memberikan kualitas dengan harga yang pantas kepada konsumen.
3. Memanfaatkan semua potensi yang dimiliki seefisien dan selektif mungkin.
4. Menjadikan CV. Avis Car Accessories sebagai perusahaan terdepan di bidangnya.

2.1.4 Struktur Organisasi Perusahaan

Struktur organisasi merupakan pola hubungan antara bagian perusahaan atau meggambarkan dengan jelas pemisah kegiatan pekerja antara bagian yang satu dengan yang lain dalam suatu perusahaan. Gambar 2.1 menjelaskan struktur organisasi yang ada di CV. Avis Car Accessories.



Gambar 2. 1 Struktur Organisasi CV. Avis *Car Accessories*

2.1.5 Deskripsi kerja

Pada sub-bab ini akan menjelaskan tentang deskripsi kerja CV. Avis Car Accessories.

1. Pimpinan/Pemilik

a. Mengontrol semua kegiatannusaha yang terjadi di CV. Avis Car Accessories.

b. Menyusun rencana dan evaluasi kegiatan serta anggaran di CV. Avis Car Accessories.

2. Bagian Administrasi

a. Melakukan / Mencatat penjualan kepada pelanggan.

b. Mencatat pengiriman barang.

c. Membuat laporan penjualan.

d. Mencatat penerimaan pembayaran piutang pelanggan.

3. Bagian Gudang

a. Mencatat barang yang masuk dan yang keluar di gudang.

b. Memberikan laporan barang yang kurang laku terjual serta barang yang sudah hampir habis stoknya.

4. Bagian Operasional

a. Mengantarkan barang-barang pesanan.

b. Melakukan promosi produksi.

c. Melakukan penawaran barang-barang baru.

5. Bagian Mekanik

a. Melakukan perbaikan atau memasangkan variasi mobil permintaan konsumen ke kendaraan konsumen.

b. Service ringan, berat dan perawatan mesin.

2.2 Landasan Teori

Landasan teori merupakan penjelasan tentang teori-teori pendukung dalam penyelesaian tugas akhir ini.

2.2.1 Pengertian Data

Menurut Inmon, data adalah sebuah rekaman dari fakta-fakta, konsep-konsep atau intruksi-intruksi pada media penyimpanan untuk komunikasi, perolehan dan pemrosesan dengan cara otomatis dan presentasi sebagai informasi yang dapat dimengerti oleh manusia.

Menurut Laudon & Laudon, data adalah fakta-fakta yang mewakili kejadian-kejadian yang berlangsung dalam organisasi atau lingkungan fisik sebelum ditata dan diatur kedalam bentuk yang dapat dipahami dan digunakan orang [5].

Dari dua teori tersebut dapat disimpulkan bahwa data merupakan kumpulan fakta yang secara umum tidak dapat digunakan karena belum diolah yang terdapat pada media penyimpanan untuk melalui pemrosesan sehingga dapat menyajikan informasi yang dapat dimengeti oleh manusia.

2.2.2 Pengertian Informasi

Informasi adalah data yang telah diolah menjadi sebuah bentuk yang berarti bagi penerimanya dan bermanfaat dalam pengambilan saat ini atau saat mendatang. Dapat disimpulkan bahwa:

1. Informasi bermuara pada data.
2. Memberkan suatu nilai tambah atau pengetahuan bagi yang menggunakannya.
3. Dapat dignakan untuk pengambila keputusan.

2.2.3 Basis Data

Basis Data adalah kumpulan data yang umumnya menjabarkan aktivitas-aktivitas dari suatu atau lebih dari satu organisai yang terkait. Basis Data adalah sebuah cara mendokumentasikan berbagai macam data yang kemudian di manajemen dengan sebuah sistem untuk kemudian disimpan dalam sebuah media penyimpanan. Dalam basis data, data yang ada tidak hanya akan disimpan begitu saja melainkan akan dikelola dengan sistem pengaturan basis data yang sering

disebut *Database Management Sistem* (DBMS). Dengan begitu pengaksesan data dapat dilakukan dengan mudah dan cepat oleh pengguna [5].

Keuntungan basis data:

1. Mereduksi redudansi.
2. Data dapat di-*share* antar aplikasi.
3. Standarisasi dapat dilakukan.
4. Batasan *security* dapat diterapkan.
5. Mengelola integritas data.
6. Independensi data.

2.2.4 Data Warehouse

Data *warehouse* merupakan kumpulan informasi *logical* yang dikumpulkan dari berbagai operasional *database* yang berguna untuk mendukung aktivitas para analis dalam pekerjaan mengambil keputusan. Penyajian data warehouse bersifat multidimensional data, yaitu *data warehouse* terdiri dari beberapa layer-layer, kolom dan baris. Layer-layer di dalam suatu *data warehouse* menunjukkan informasi yang disampaikan pada dimensi yang berbeda.

Data *warehouse* juga dapat dikatakan sebagai tempat penyimpanan data historis yang berasal dari berbagai departemen, organisasi atau perusahaan yang bertujuan untuk mengolah data-data tersebut untuk menghasilkan sebuah informasi yang efisien dan efektif sesuai dengan keinginan pengguna serta mempermudah pihak analis untuk pengambilan hasil keputusan. Data *warehouse* merupakan kumpulan dari *database* yang berbeda-beda, *data warehouse* mengombinasikan informasi dengan cara meringkas dan mengelompokkan. *Data warehouse* merupakan metode dalam perancangan basis data yang dapat menunjang DSS (*Decision Support System*) dan EIS (*Executive Information System*).

Menurut W.H. Inmon, data *warehouse* merupakan kumpulan koleksi data yang mempunyai sifat berorientasi subjek, *integrated*, *time-variant*, dan bersifat

tetap/tidak berubah (*non-volatile*) dari koleksi data dalam mendukung proses pengambilan keputusan manajemen [6].

Ada empat tingkat data di dalam arsitektur basis data yang dapat dilihat pada Gambar 2.2 yaitu sebagai berikut :

1. Tingkat operasional

   Pada tingkatan ini data yang diolah merupakan data sehari-hari, terperinci, dan berorientasi aplikasi.

2. Tingkat *Data Warehouse*

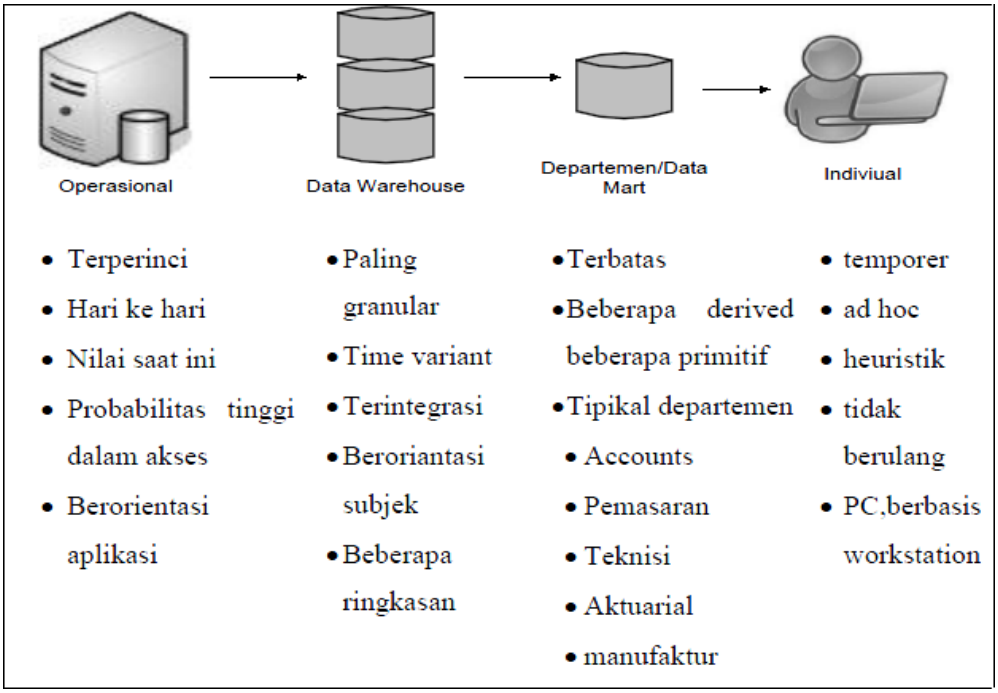
   Merupakan kumpulan koleksi data yang mempunyai sifat berorientasi subjek, *integrated*, *time-variant*, dan *non-volatile*. Data warehouse merupakan kumpulan dari beberapa data *mart*.

3. Tingkat Departemen/*data mart*

   Data Mart merupakan bentuk sederhana dari sebuah data *warehouse*, data *mart* hanya fokus pada sebuah subjek tunggal atau suatu area fungsional seperti bagian penjualan, produksi, keuangan dan lainnya.

4. Tingkat individual

   Data individual bersifat temporer dan hanya dalam ruang lingkup kecil. Didukung oleh perangkat komputer di mana proses EIS yang berlangsung hanya berjalan di dalam level individu.



Gambar 2. 2 Arsitektur Data *warehouse*

2.2.4.1 Karakteristik Data Warehouse

Menurut W.H. Inmon, data warehouse memiliki beberapa karakteristik yaitu [6] :

1. *Subject Oriented*

Data warehouse berorientasi subjek artinya *data warehouse* di desain untuk menganalisa data berdasarkan subjek-subjek tertentu dalam organisasi, bukan pada proses atau fungsi pada aplikasi tertentu.

2. *Integrated*

Data warehouse menyimpan data-data yang berasal dari beberapa sumber terpisah ke dalam suatu format yang konsisten dan saling terhubung satu dengan yang lainnya. Dengan demikian data tidak bisa dipecah – pecah karena data yang ada merupakan satu kesatuan yang menunjang keseluruhan konsep *data warehouse* itu sendiri. Integrasi sumber data dapat dilakukan dengan berbagai cara seperti konsisten dalam penamaan *variable*, konsisten dalam ukuran *variable*, konsisten dalam struktur pengodean, dan konsisten dalam atribut fisik dari data.

3. *Time variant*

Seluruh data pada data *warehouse* dapat dikatakan akurat atau valid pada rentang waktu tertentu.

4. *Non-Volatile*

Non-volatile berarti data tidak berubah, maksudnya data yang ada pada sebuah data *warehouse* tidak di *update* secara *real time* akan tetapi akan di *refresh* dari sistem operasional secara reguler. Data yang baru selalu ditambahkan sebagai suplemen bagi *database* itu sendiri dari pada sebagai sebuah perubahan. *Database* tersebut secara terus-menerus menyerap data tersebut, kemudian akan disatukan dengan data sebelumnya.

2.2.4.2 Keuntungan Data Warehouse

Menurut Conolly dan Begg, data *warehouse* dapat memberikan keuntungan bagi perusahaan, diantaranya:

1. Tingkat pengembalian investasi yang tinggi

Sebuah organisasi harus menangani sumber daya dalam jumlah besar untuk memastikan implementasi *data warehouse* yang berhasil dan biayanya bias sangat bervariasi. Berdasarkan penelitian dari *International* dan *Corporation* pada 1996 rata-rata tingkat pembelian investasi *data warehouse* dalam tiga tahun mencapai 40%.

2. Keuntungan kompetitif

Pengembalian dari investasi yang tinggi bagi perusahaan yang mengimplementasikan *data warehouse* dengan berhasil akan memberikan mereka keuntungan kompetitif. Keuntungan tersebut didapat dengan mengizinkan pembuat keputusan mengakses data yang sebelumnya tidak tersedia, tidak diketahui dan informasi yang belum bermanfaat, contohnya trend dan permintaan.

3. Meningkatkan produktivitas para pembuat keputusan perusahaan

Data warehouse meningkatkan produktivitas dari pembuat keputusan perusahaan dengan menciptakan *database* yang terintegrasi, konsisten, berorientasi subjek dan data historis. Dengan merubah data menjadi informasi

yang berarti, *data warehouse* memungkinkan manajeruntuk melakukan analisis dengan lebih akurat dan konsisten.

2.2.5 Data Mart

Data *mart* merupakan suatu bagian dari data *warehouse* yang berguna untuk mendukung pembuatan laporan dan analisis data pada suatu bagian operasional departemen dalam sebuah perusahaan [2]. Data *mart* berisi informasi yang relevan bagi pengguna yang nantinya dapat mempermudah dalam hal pengambilan keputusan. Berdasarkan sumber data yang digunakan, data *mart* terbagi menjadi dua bagian, yaitu:

1. Dependen Data *Mart*

   Dependen data *mart* mengambil data dari sebuah data *warehouse* yang sudah dibuat. Dependen data *mart* biasanya dibangun untuk mencapai peningkatan kinerja dan ketersediaan, kontrol yang lebih baik, dan biaya telekomunikasi yang lebih rendah yang dihasilkan dari akses lokal data yang relevan dengan suatu departemen tertentu.

2. Independen Data *Mart*

   Independen data *mart* merupakan sistem mandiri yang dibangun dengan membuat data langsung dari sumber operasional atau sumber eksternal dari data. Penciptaan independen *mart* sering didorong berdasarkan kebutuhan untuk memiliki sebuah solusi dalam waktu yang lebih singkat.

Data *mart* dan data *warehouse* memiliki perbedaan, yaitu:

Tabel 2.1 Perbedaan Data Mart dan Data Warehouse

Data Warehouse	***Data Mart***
Merujuk pada keseluruhan organisasi	Merujuk pada departemen tertentu
Gabungan dari data *mart*	Sebuah proses bisnis
Data diterima dari staging area	Star join (fakta dan dimensi)

Data Warehouse	*Data Mart*
Di-Query disumber tampilan	Teknologi yang maksimal untuk akses dan analisis data
Membentuk pandangan data untuk organisasi	Membentuk pandangan data per departemen
Diorganisasikan dalam ER-model	-

2.2.5.1 Jenis-jenis Data Source Untuk Data *Mart*

Data source dapat dibedakan menurut beberapa jenis kebutuhannya, dan berdasarkan sumber data yang diperoleh. Berikut ini adalah jenis-jenis dari *data source :*

1. *On-Line Transaction Processing* (OLTP)

   Merupakan sebuah sistem yang memfasilitasi dan mengelola aplikasi yang berorientasi transaksi. Biasanya untuk data *entry* atau pengambilan proses transaksi.

2. *Sistem Legacy*

   Sistem yang sudah *out-of-date* tetapi data-datanya masih digunakan untuk keperluan *historical*.

3. *External Data*

   Data yang berasal dari luar perusahaan.

4. *Middleware*

   Sebuah aplikasi yang menjembatani antara aplikasi yang berbeda.

5. *Source-less Data*

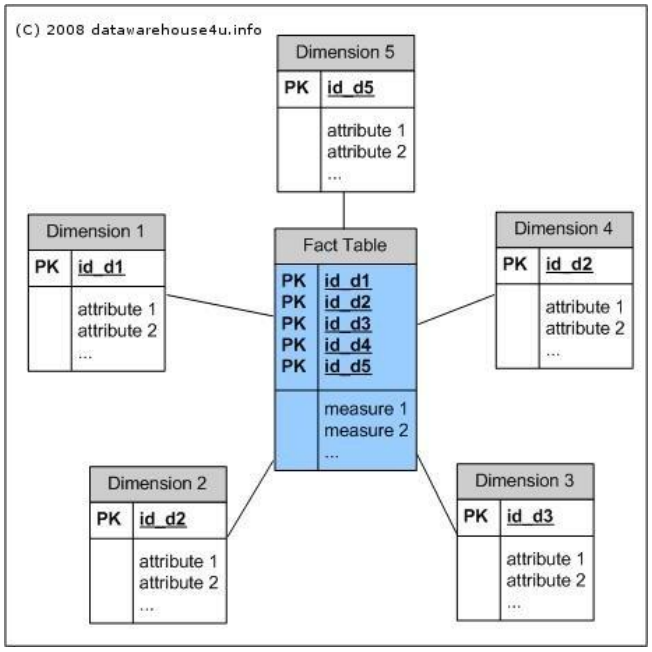
   Data yang sudah tidak ada lagi sumber datanya.

2.2.5.2 Model Dimensional Data *Mart*

Ada beberapa model skema yang terdapat pada pemodelan data *mart*, yaitu skema star, skema *snowflake*, dan skema *fact constellation*. Penjelasan dari masing-masing model skema adalah sebagai berikut :

1. Skema *Star*

Skema *star* terdiri dari dua macam tabel, yaitu tabel fakta dan tabel dimensi. Tabel fakta mengandung fakta terhadap data kuantitatif mengenai sebuah bisnis seperti jumlah unit terjual, jumlah pemesanan dan lainnya. Tabel dimensi berisi data deskriptif mengenai subjek bisnis. Tabel dimensi biasanya sebagai sumber atribut yang digunakan untuk mengategorikan, atau meringkas fakta dalam *query*, *report*, atau grafik [2]. Untuk lebih jelasnya dapat dilihat pada Gambar 2.3 berikut.

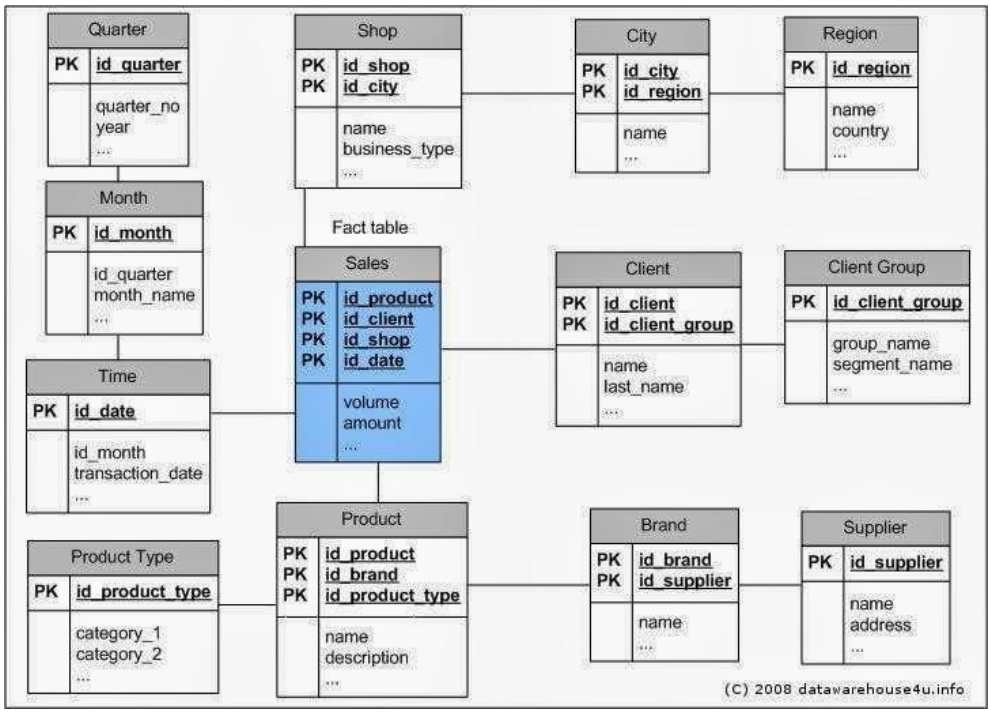


Gambar 2. 3 Model *Star*

2. Skema *Snowflake*

Skema *snowflake* merupakan metode normalisasi tabel dimensi dalam skema *star* dimana setelah semua tabel dimensi telah benar-benar dinormalisasikan. Prinsip dari skema ini adalah menormalisasi tabel dengan menghapus atribut

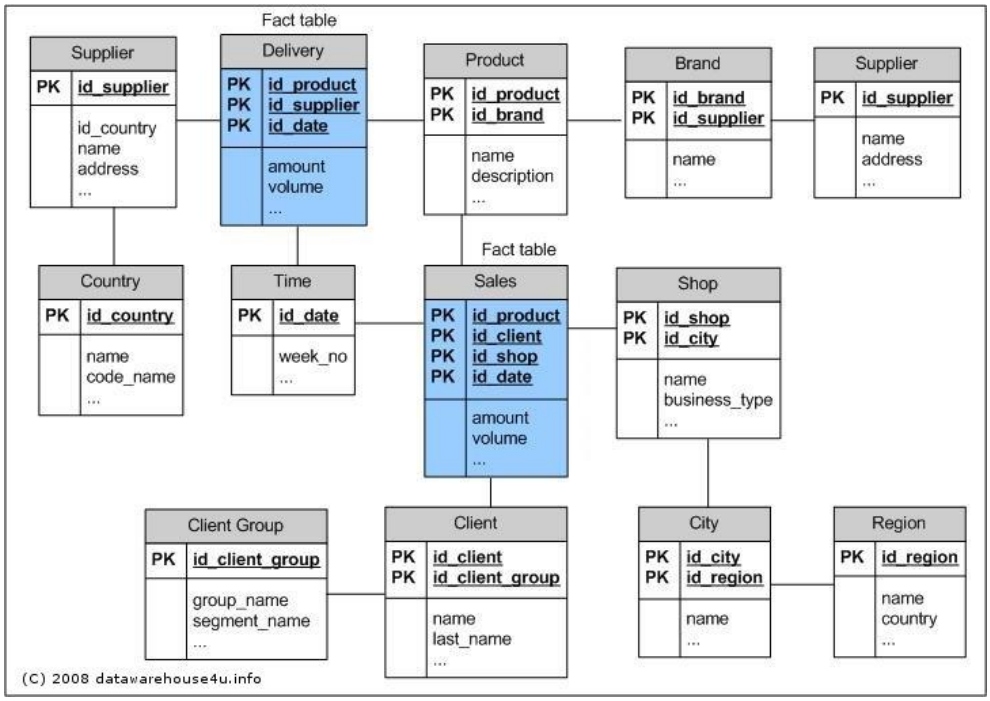
dengan kardinalitas rendah dan membentuk tabel terpisah. Alasan melakukan *snowflaking* juga adalah untuk penghematan ruang penyimpanan dalam satu tabel dan penelusuran atribut lebih terperinci dalam suatu tabel dimensi [2]. Untuk lebih jelasnya dapat dilihat pada Gambar 2.4 berikut.



Gambar 2. 4 Model *Snowflake*

3. Skema *Constellation*

Skema ini lebih kompleks dibanding dengan sekma *star* dan skema *snowflake* karena skema ini berisi dari dua atau lebih tabel fakta. Skema ini memungkinkan suatu tabel dimensi berhubungan dengan banyak tabel fakta. Skema *Fact Constellation* sangat fleksibel, namun terkadang menjadi susah dalam pengaturan dan *support*. Kerugian utama skema ini adalah desain lebih rumit karena banyak varian agregasi yang harus dipertimbangkan, selain itu juga untuk menjawab *single query* mungkin butuh *multiple* SQL *statement* [3]. Untuk lebih jelasnya dapat dilihat pada Gambar 2.5 berikut.



Gambar 2. 5 Model *Constellation*

2.2.5.3 Stagging Area

Data *staging* merupakan suatu proses yang biasa disebut dengan proses ETL (*Extract, Transform, and Load*). Data *staging* bertujuan untuk menentukan data apa saja yang digunakan sebagai data fakta dan data dimensi yang ada dalam *dimensional model*, cara mengekstraksinya, mengkondisikan skema data dengan kebutuhan, dan proses memindahkan data dari sumber data ke *dimensional* model [3]. Adapun penjelasan dari proses ETL adalah sebagai berikut :

1. *Extract*

   Proses ini merupakan tahapan untuk menentukan tabel beserta *field* yang akan di esktrak dari sumber data ke dalam *dimensional model* [3].

2. *Transform*

   Proses ini merupakan proses mengkondisikan bentuk skema data, baik itu tabel ataupun atributnya untuk disesuaikan dengan kebutuhan. Proses *transform* yang dilakukan adalah *cleaning* dan *conditioning* [3].

a. *Cleaning*

Proses *cleaning* membersihkan data-data yang tidak perlu dari tabel yang telah di ekstrak, yaitu menghilangkan *field* yang tidak terpakai.

b. *Conditioning*

Proses ini merupakan proses untuk melakukan pengkondisian bentuk skema data yang terdiri dari pengkondisian atribut waktu pada tabel dan pengkondisian nama tabel sehingga dapat digunakan oleh *dimensional model*.

3. *Load*

Proses ini merupakan tahapan akhir dari proses data *staging*, di mana data fakta dan dimensi dimuat ke dalam *dimensional model*. Teknik yang dilakukan dalam *delivering* data tergantung kebutuhan bisnis, diantaranya adalah teknik *update*, di mana teknik ini akan langsung memperbaharui data yang tersimpan dalam *dimensional model* tanpa menghilangkan atau mengubah data yang sudah ada sebelumnya [3].

2.2.5.4 Normalisasi

Normalisasi adalah suatu proses yang digunakan untuk menentukan pengelompokan 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 redudansi sedikit mungkin
2. Memungkinkan baris-baris dalam relasi disisipkan, dimodifikasi, dan dihapus tanpa menimbulkan kesalahan atau ketidakkonsistenan.

Normalisasi sendiri dilakukan melalui sejumlah langkah. Setiap langkah berhubungan dengan bentuk normal (normal form) tertentu. Dalam hal ini yang disebut bentuk normal adalah suatu keadaan relasi yang dihasilkan oleh penerapan aturan-aturan sederhana yang berhubungan dengan dependensi fungsional terhadap relasi tersebut. Yang dimaksud dengan aturan-aturan tersebut dan juga istilah dependensi fungsional akan dibahas belakangan. Sekedar perlu diketahui terlebih dulu bahwa bentuk normal dalam normalisasi dapat berupa:

1. Bentuk normal pertama (1NF / First normal form)
2. Bentuk normal pertama (2NF / Second normal form)
3. Bentuk normal kedua (3NF / Third normal form)
4. Bentuk normal Boyce-Codd (BCNF / Boyce-Codd normal form)
5. Bentuk normal keempar (4NF / Fourth normal form)
6. Bentuk normal kelima (5NF / Fifth normal form)

Masing-masing bentuk normal akan dibahas tersendiri. Adapun hubungan keenam bentuk normal tersebut [6].

2.2.6 OLAP (On-Line Analytical Processing)

OLAP (*On-line Analytical Processing*) adalah satu cara untuk mengolah data pada sebuah data *warehouse*. OLAP memberikan jawaban terhadap *query analytic* untuk data yang bersifat multidimensional. OLAP menyediakan cara untuk menampilkan data multidimensional yang ada dalam data *mart* atau data *warehouse*, dengan OLAP dapat dibuat *cube* yang mengorganisasikan data dan membuat *summary* data untuk *query* yang efisien [6]. Teknik ini ada berbagai macam, di antaranya adalah :

1. *Drill-down and roll-up*

Drill-down and roll-up adalah operasi untuk melihat data global atau detail di sepanjang level hierarki dimensi. *Roll-up* untuk melihat data secara global atau rangkuman (*summary*). *Drill-down* memandu pengguna untuk memperoleh data yang lebih detail. *Drill-down* ini biasa digunakan untuk menjawab pertanyaan atas suatu kasus tertentu. Misalnya untuk menjawab pertanyaan ketika sebuah *summary number* (rata-rata atau jumlah) di bawah atau di atas harapan.

2. *Slice and dice*

Slice and dice adalah operasi untuk melihat data sebagai visualisasi dari kubus. Dengan *slice and dice* pengguna dapat melihat data dari berbagai perspektif. Pengguna dapat mengekstrak bagian dari data agregasi dan dapat memeriksa dengan detail berdasarkan dimensi-dimensi yang diinginkan. Data agregasi merupakan data pra-perhitungan dalam bentuk rangkuman data (data *summarized*) sehingga *query* pada kubus lebih cepat. *Slice* memotong kubus

sehingga dapat memfokuskan pada perspektif yang spesifik pada suatu dimensi. Sedangkan *dice* memberikan kemampuan untuk melihat pemilihan data pada dua dimensi atau lebih, yaitu dengan merotasi *cube* pada perspektif yang lain sehingga pengguna dapat melihat lebih spesifik terhadap data yang di analisis.

2.2.7 OLTP (On-Line Transaction Processing)

On-line Transaction Processing merupakan suatu pemrosesan yang menyimpan data mengenai kegiatan operasional transaksi sehari-hari atau singkatnya merupakan segala data penanganan dalam proses transaksi sehari-hari yang terjadi [6].

Ciri-ciri umum sistem OLTP adalah sebagai berikut:

1. Mendukung jumlah pengguna dalam jumlah yang banyak dan sering menambah dan mengubah data transaksi.
2. Mengandung data dalam jumlah besar, termasuk di dalamnya validasi data transaksi.
3. Memiliki struktur yang kompleks dan rumit.
4. Diarahkan secara maksimal untuk melayani aktivitas transaksi harian.
5. Menyediakan teknologi infrastruktur yang mendukung operasional transaksi data dalam perusahaan.

2.2.8 Visualisasi Data

Bentuk visualisasi data yang umum digunakan adalah sebagai berikut :

1. *Pie Chart*

Pie Chart adalah sebuah tampilan deskriptif sederhana dari data, dimana data yang ditampilkan hanya jumlah dari total yang diberikan. *Pie Chart* mungkin adalah sebuah cara paling ilustratif untuk menampilkan kuantitas sebagai presentasi dari total yang diberikan. Total area dari sebuah *pie chart* mempresentasikan 100% dari kuantitas (jumlah dari nilai variabel pada seluruh kategori). Ukuran dari setiap potongan adalah presentasi dari total yang

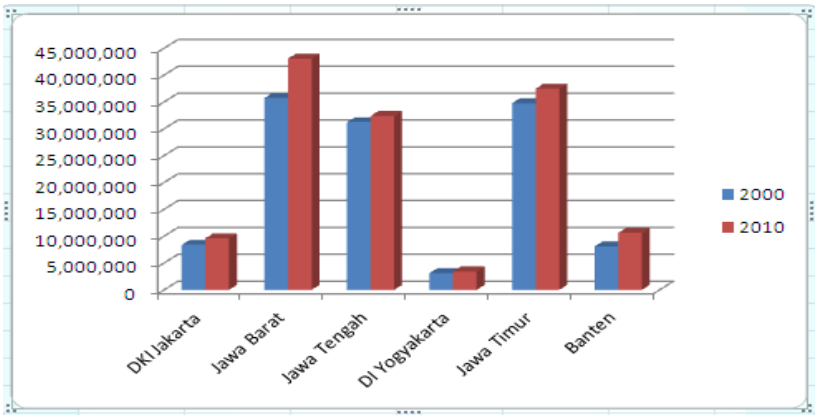
dipresentasikan oleh masing-masing kategori yang dapat dilihat pada Gambar 2.6 berikut.



Gambar 2. 6 Contoh *Pie Chart*

2. *Bar Chart*

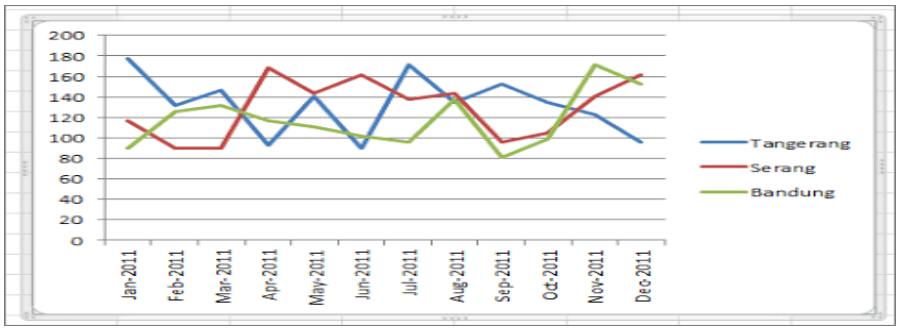
Bar chart sering digunakan untuk menunjukan data berdasarkan kategori tertentu dimana tidak ada penekanan total presentase pada setiap kategori. *Bar chart* dapat disajikan secara vertikal maupun horizontal. Skala pengukuran adalah nominal atau ordinal. *Bar chart* dapat digunakan untuk menampilkan data kontinu seperti ukuran sepatu atau warna mata dan data diskontinu seperti tinggi badan atau berat badan. Contoh dari *bar chart* dapat dilihat pada Gambar 2.7 berikut ini.



Gambar 2. 7 Contoh *Bar Chart*

3. *Line Chart*

Line chart sering digunakan untuk menampilkan informasi dalam rangkaian titik data yang dihubungkan dengan segmen garis lurus. *Line chart* sering digunakan untuk memvisualisasikan trend data dalam interval waktu atau dalam kurun waktu tertentu. Contoh dari *line chart* dapat dilihat pada Gambar 2.8 berikut ini.



Gambar 2. 8 Contoh *Line Chart*

2.2.9 Microsoft SQL Server

Microsoft SQL Server adalah sebuah sistem manajemen basis data relasional (RDBMS) produk *Microsoft*. Bahasa *query* utamanya adalah *Transact-SQL* yang merupakan implementasi dari *SQL* standar ANSI/ISO yang digunakan oleh *Microsoft* dan *Sybase*. Umumnya *SQL Server* digunakan di sunia bisnis yang memiliki basis data yang berskala kecil sampai dengan menengah, tetapi kemudian berkembang dengan digunakannya *SQL Server* pada basis data besar. Pengguna yang menggunakan *SQL Server* dapat menyimpan banyak data dan dapat untuk diimplementasi untuk kepentingan bisnis dan perusahaan [7].

2.2.10 SSIS (*SQL Server Integration Service*)

SSIS (*SQL Server Integration Services*) adalah suatu platform untuk membangun sistem yang handal untuk integrasi data, *extraction*, *transformation*, dan *loading* yang digunakan pada *data warehousing*. SSIS menawarkan solusi dalam menghadapi permasalahan data integrasi. Selain itu, *tools* ini membatu untuk menigkatkan efisiensi waktu pembuatan [8].

Arsitektur *SQL Server Integration Services* secara umum berisi berbagai macam komponen, diantaranya yaitu :

1. *SSIS Deginer*. Merupakan *tools* yang digunakan untuk membuat dan mengatur paket *integration service*. Pada *SQL Server 2012*, *tools* ini sudah terintegrasi dengan *Visual Studio 2010* yang merupakan bagian *project Bussiness Intelegence*.
2. *Runtime Engine*. Komponen ini berguna untuk menjalankan semua paket SSIS yang sudah dibuat.
3. *Task* dan *executable binary*.
4. *Data Flow Engine* dan *Data Flow*. Komponen *data flow* merupakan enkapsulasi *data flow engine* yang menyediakan *buffer* didalam memori dan bertugas memindahkan data dari sumber data ke tujuan data. Sedangkan *data flow* merupakan sumber data, tujuan data, dan transformasi.
5. *Integration Services service*. Memungkinkan *SQL Server Management Studio* dapat digunakan untuk melakukan monitoring paket SSIS dan mengatur SSIS *storage* yang digunakan.
6. *SQL Server Import* dan *Export Wizard*. *Tools* ini digunakan untuk melakukan *copy* data dari sumber ke tujuan data.

2.2.11 Analisis Berorientasi Objek

Analisis berorientasi objek atau *Object Oriented Analysis* adalah tahapan untuk menganalisis spesifikasi atau kebutuhan akan sistem yang akan dibangun dengan konsep berorientasi objek, apakah benar kebutuhan yang ada dapat diimplementasikan menjadi sebuah sistem berorientasi objek. *Object Oriented Analysis* biasanya menggunakan kartu CRC *(Component, Responsibility, Collaborator)* untuk membangun kelas-kelas yang akan digunakan atau menggunakan UML (*Unified Modeling Language)* pada bagian diagram *use case*, diagram kelas dan diagram objek [9].

2.2.12 Unified Modeling Language

UML (*Unified Modelling Language*) adalah salah satu alat bantu yang sangat handal di dunia pengembangan sistem yang berorientasi objek. Hal ini disebabkan karena UML menyediakan bahasa pemodelan visual yang memungkinkan bagi pengembang untuk membuat cetak biru atas visi mereka dalam bentuk yang baku, mudah untuk dimengerti serta dilengkapi dengan mekanisme yang efektif untuk berbagi (*sharing*) dan mengkomunikasikan rancangan mereka yang lain [10].

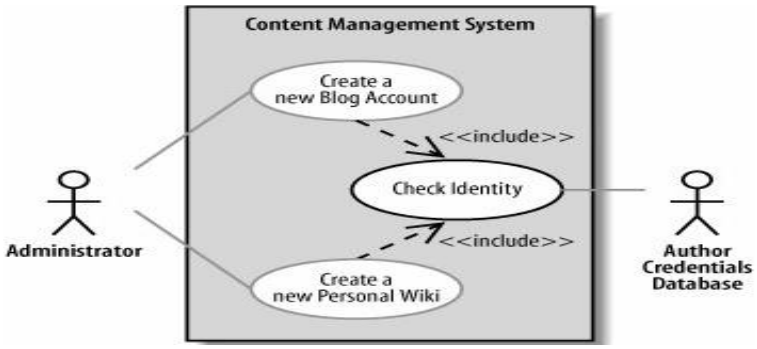
UML merupakan kesatuan dari bahasa pemodelan yang dikembangkan oleh Booch, *Object Modelling Technique* (OMT) dan *Object Oriented Software Engineering* (OOSE). Metode Booch dari Grady Booch sangat terkenal dengan nama metode *Design Object Oriented*. Metode ini menjadikan proses analisis dan design ke dalam empat tahapan iteratif, yaitu identifikasi kelas-kelas dan objek-objek, identifikasi semantik dari hubungan objek dan kelas tersebut, perincian interface dan implementasi. Keunggulan metode Booch adalah pada detail kayanya dengan notasi dan elemen. Pemodelan OMT yang dikembangkan oleh Rumbaugh didasarkan pada analisis terstruktur dan pemodelan *entity-relationship*. Tahapan utama dalam metodologi ini adalah analisis, design sistem, design objek dan implementasi. Keunggulan metode ini adalah dalam penotasian yang mendukung semua konsep OO. Metode OOSE dari Jacobson lebih memberi penekanan pada *use case*. OOSE memiliki tiga tahapan yaitu membuat model *requirement* dan analisis, design dan implementasi, dan model pengujian (*test model*). Keunggulan metode ini adalah mudah dipelajari karena memiliki notasi yang sederhana namun mencakup seluruh tahapan dalam rekayasa perangkat lunak.

Dengan UML, metode Booch, OMT dan OOSE digabungkan dengan membuang elemen-elemen dari metode lain yang lebih efektif dan elemen-elemen baru yang belum ada pada metode terdahulu sehingga UML lebih ekspresif dan seragam daripada metode lainnya.

2.2.12.1 Use Case Diagram

Use Case adalah deskripsi 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. Urutan langkah-langkah yang menerangkan antara pengguna dan sistem disebut skenario. Setiap skenario mendeskripsikan urutan kejadian. Setiap urutan diinisialisasi oleh orang, sistem lain, perangkat keras atau urutan waktu. Dengan demikian secara singkat biasa dikatakan *use case* adalah serangkaian skenario yang digabungkan bersama-sama oleh tujuan umum pengguna [10].

Berikut ini adalah contoh Gambar 2.9 *Use Case Diagram*:



Gambar 2. 9 Use Case Diagram

2.2.12.2 Use Case Scenario

Sebuah diagram yang menunjukkan kasus penggunaan dan aktor dapat menjadi titik awal yang bagus, tetapi tidak memberikan cukup detail untuk desainer sistem untuk benar-benar memahami persis bagaimana kekhawatiran sistem akan dipenuhi. *Use Case* Skenario adalah cara terbaik untuk mengungkapkan informasi penting ini dalam bentuk penggunaan description setiap kasus berbasis teks harus disertai dengan satu [10].

Berikut ini adalah contoh Gambar 2.10 *Use Case Scenario*:

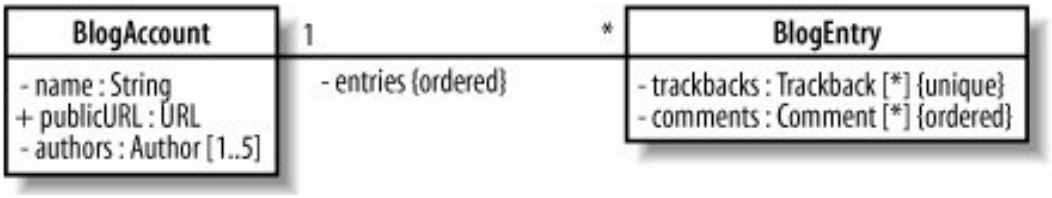
Use case name	Create a new Blog Account	
Related Requirements	Requirement A.1.	
Goal In Context	A new or existing author requests a new blog account from the Administrator.	
Preconditions	The system is limited to recognized authors and so the author needs to have appropriate proof of identity.	
Successful End Condition	A new blog account is created for the author.	
Failed End Condition	The application for a new blog account is rejected.	
Primary Actors	Administrator.	
	Secondary Actors	Author Credentials Database.
	Trigger	The Administrator asks the CMS to create a new blog account.
Main Flow	Step	Action
	1	The Administrator asks the system to create a new blog account.
	2	The Administrator selects an account type.
	3	The Administrator enters the author's details.
	4	The author's details are verified using the Author Credentials Database.
	5	The new blog account is created.
	6	A summary of the new blog account's details are emailed to the author.
Extensions	Step	Branching Action
	4.1	The Author Credentials Database does not verify the author's details.
	4.2	The author's new blog account application is rejected.

Gambar 2. 10 *Use Case Scenario*

2.2.12.3 Class Diagram

Class Diagram adalah diagram yang digunakan untuk menampilkan beberapa kelas serta paket-paket yang ada dalam sistem/perangkat lunak yang sedang kita kembangkan. Diagram kelas memberikan gambaran/diagram statis tentang sistem/perangkat lunak dan relasi-relasi yang ada di dalamnya [10].

Berikut ini adalah contoh Gambar 2.11 *Class Diagram*:



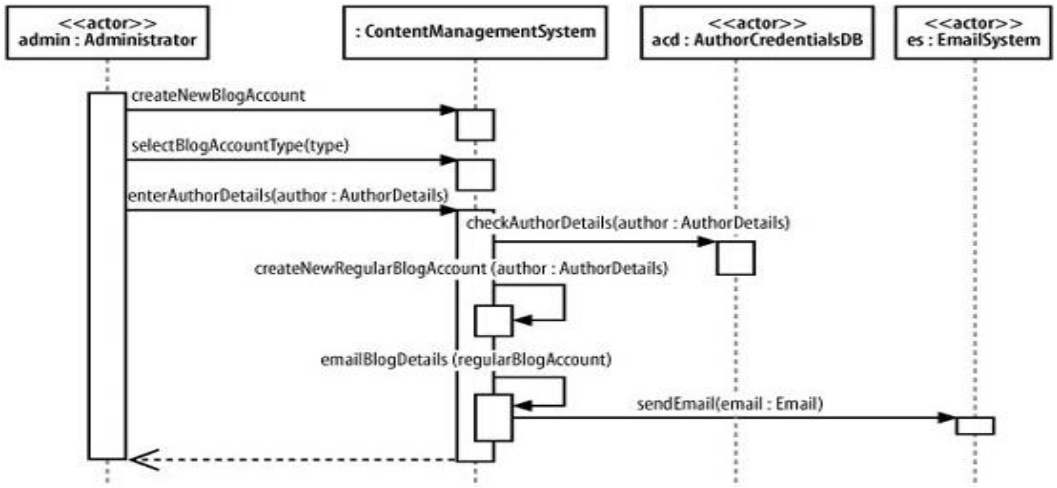
Gambar 2. 11 *Class* Diagram

2.2.12.4 Sequence Diagram

Sequence Diagram digunakan untuk menggambarkan perilaku pada sebuah skenario. Diagram ini menunjukkan sejumlah contoh objek dan *message* (pesan) yang diletakkan diantara objek-objek ini dalam *use case* [10].

Komponen utama *sequence diagram* terdiri atas objek yang dituliskan dengan kotak segiempat bernama. *Message* diwakili oleh garis dengan tanda panah dan waktu yang ditunjukkan dengan *progress* vertikal.

Berikut ini adalah contoh Gambar 2.12 *Sequence Diagram*:

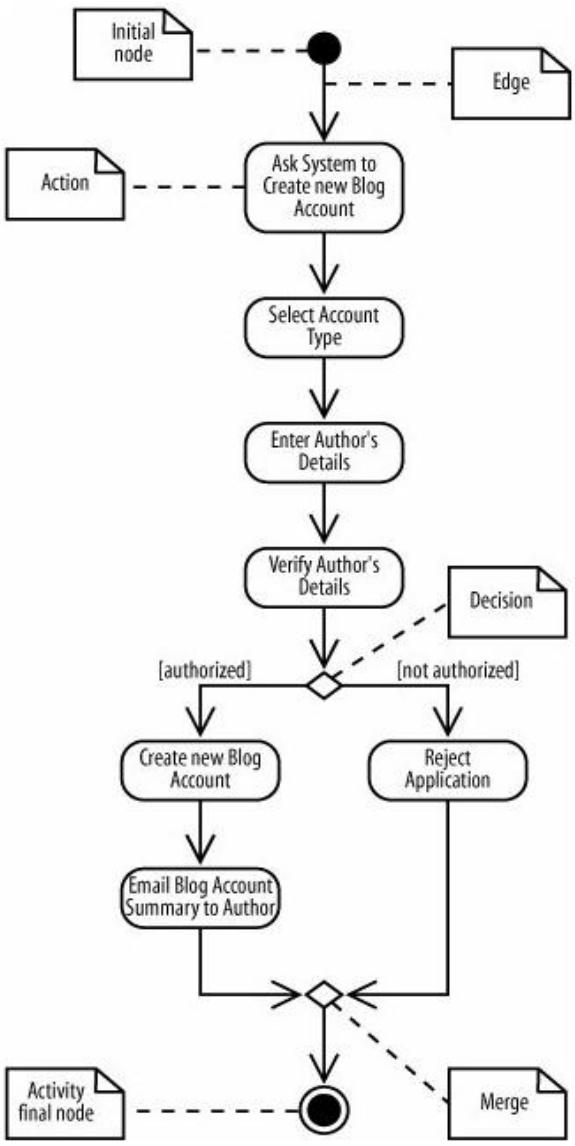


Gambar 2. 12 *Sequence* Diagram

2.2.12.5 *Activity Diagram*

Activity Diagram adalah teknik untuk mendeskripsikan 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 mendukung perilaku paralel sedangkan *flowchart* tidak bisa [10].

Berikut ini adalah contoh Gambar 2.13 *Activity Diagram*:



Gambar 2. 13 Activity Diagram