# BAB III

# LANDASAN TEORI

Pada bab ini akan dijelaskan dasar-dasar teori yang berhubungan dengan permasalahan yang dibahas, dan menjelaskan system yang digunakan pada kerja praktik ini. Adapun teori-teori yang digunakan sebagai landasan pemikiran dalam kerja praktik ini adalah sebagai berikut:

## 1.1. Konsep Dasar Aplikasi

Aplikasi adalah suatu sub kelas perangkat lunak komputer yang memanfaatkan kemampuan komputer langsung untuk melakukan suatu tugas yang diinginkan pengguna. Biasanya dibandingkan dengan perangkat lunak sistem yang mengintegrasikan berbagai kemampuan komputer, tapi tidak secara langsung menerapkan kemampuan tersebut untuk mengerjakan suatu tugas yang menguntungkan pengguna. Contoh utama perangkat lunak aplikasi adalah pengolah kata, lembar kerja, dan pemutar media. beberapa aplikasi yang digabung bersama menjadi suatu paket kadang disebut sebagai suatu paket atau suite aplikasi (*application suite*). Contohnya adalah *Microsoft Office* dan *OpenOffice.org*, yang menggabungkan suatu aplikasi pengolah kata, lembar kerja, serta beberapa aplikasi lainnya. Aplikasi-aplikasi dalam suatu paket biasanya memiliki antarmuka pengguna yang memiliki kesamaan sehingga memudahkan pengguna untuk mempelajari dan menggunakan tiap aplikasi. Sering kali, mereka memiliki kemampuan untuk saling berinteraksi satu sama lain sehingga menguntungkan pengguna. Contohnya, suatu lembar kerja dapat dibenamkan

INSTITUT BISNIS & INFORMATIKA stikom SURABAYA

dalam suatu dokumen pengolah kata walaupun dibuat pada aplikasi lembar kerja yang terpisah.

Definisi aplikasi menurut para ahli adalah sebagai berikut:

1. Menurut Hendrayudi, aplikasi adalah kumpulan perintah program yang dibuat untuk melakukan pekerjaan-pekerjaan tertentu.
2. Menurut Hengky W. Pramana, aplikasi adalah suatu unit perangkat lunak yang dibuat untuk melayani kebutuhan akan beberapa aktivitas seperti sistem perniagaan, *game* pelayanan masyarakat, periklanan, atau semua proses yang hampir dilakukan manusia.
3. Menurut Harip Santoso, aplikasi adalah suatu kelompok file (*Form*, *Class*, *Report*) yang bertujuan untuk melakukan aktivitas tertentu yang saling terkait.
4. Menurut Ibisa, aplikasi adalah alat bantu untuk mempermudah dan mempercepat proses pekerjaan dan bukan merupakan beban bagi penggunanya.

INSTITUT BISNIS & INFORMATIKA stikom SURABAYA

## 1.2. Konsep Dasar Persediaan

Definisi persediaan dalam buku yang berjudul Akuntansi Untuk Bisnis Dan Dagang, persediaan adalah aktiva yang tersedia untuk diajual dalam kegiatan normal persediaan. Pada bisnis manufaktur, persediaan meliputi bahan mentah, barang dalam proses produksi, barang jadi (Suharli, 2006).

Persediaan adalah pos-pos aktiva yang dimiliki oleh perusahaan untuk dijual dalam operasi bisnis normal, atau barang yang akan dikonsumsi dalam membuat barang yang akan dijual (Kieso dan Weygnandt, 2007).

Menurut SAK 2009 (2009:14.2), persediaan adalah sebagai aktiva yang tersedia untuk dijual dalam kegiatan usaha biasa ; dalam proses produksi untuk penjualan tersebut ; atau dalam bentuk bahan atau perlengkapan untuk digunakan dalam proses produksi atau pemberian jasa.

Menurut Slamet Sugiri (2007:101) terdapat metode-metode penilaian persediaan yang didasarkan pada harga pokok, antara lain:

1. *First In First Out* (FIFO)

Metode FIFO atau sering disebut MPKP (Masuk Pertama Keluar Pertama) beranggapan bahwa barang yang pertama kali dibeli adalah yang pertama kali siap dan mempunyai urutan pertama untuk digunakan atau dijual. Karena itu, persediaan yang tersisa merupakan barang yang dibeli paling akhir.

2. *Last In First Out* (LIFO)

Metode ini sering disebut MTKP (Masuk Terakhir Keluar Pertama) merupakan kebalikan dari metode harga pokok FIFO. Metode LIFO didasarkan pada anggapan bahwa barang yang dibeli lebih akhir akan dijual atau dikeluarkan lebih dulu. Pada metode ini harga pokok persatuan barang-barang yang terakhir dibeli justru dibebankan kepada barang yang pertama kali dijual. Persediaan akhir dihargai dengan harga pokok pembelian yang pertama dan berikutnya. Metode LIFO, biaya dari unit yang dijual merupakan biaya pembelian paling akhir.

## 1.3. Konsep Dasar Penjualan dan Pembelian

### 1.3.1. Definisi Penjualan

INSTITUT BISNIS & INFORMATIKA stikom SURABAYA

Menurut Kotler dan Amstrong (2006:457), penjualan merupakan sebuah proses dimana kebutuhan pembeli dan kebutuhan penjualan dipenuhi, melalui antar pertukaran informasi dan kepentingan.

Jadi konsep penjualan adalah cara untuk mempengaruhi konsumen untuk membeli produk yang ditawarkan. Dalam kenyataannya penjualan mempunyai dua sistem yang biasa diterapkan oleh suatu perusahaan dagang yaitu penjualan yang dilakukan dengan cara tunai dan penjualan yang dilakukan secara kredit atau sering disebut cara angsuran.

### 1.3.2. Definisi Pembelian

Pembelian didefinisikan sebagai usaha untuk memenuhi kebutuhan barang atau jasa yang diperlukan oleh perusahaan dan diterima tepat pada waktunya dengan mutu yang sesuai serta harga yang menguntungkan.

Dalam sebuah perusahaan dagang kegiatan pembelian meliputi pembelian aktiva produktif, pembelian barang dagang, serta pembelian barang dan jasa lain dalam rangka kegiatan usaha.

INSTITUT BISNIS & INFORMATIKA stikom SURABAYA

## 3.4 Analisis dan Perancangan Sistem

### 3.4.1 Analisis Sistem

Analisis sistem yang didefinisikan oleh Al Fatta (2007: 24) adalah, sebagai bagaimana memahami dan menspesifikasi dengan detail apa yang harus dilakukan oleh sistem. Sedangkan menurut Jogiyanto (2005: 129) Analisa Sistem adalah penguraian dari suatu sistem informasi yang utuh ke dalam bagian-bagian komponennya dengan maksud untuk mengidentifikasikan dan mengevaluasi permaslahan-permaslahan, kesempatan-kesempatan, hambatan-hambatan yang

terjadi dan kebutuhan-kebutuhan yang diharapkan sehingga dapat diusulkan perbaikan-perbaikannya.

Didalam tahap analisis sistem terdapat langkah-langkah dasar yang harus dilakukan oleh analis sistem, yaitu:

1. *Identify,* yaitu mengidentifikasi masalah..
2. *Understand,* yaitu memahami kerja dari sistem yang ada.
3. *Analyze,* yaitu menganilisis sistem.
4. *Report,* yaitu membuat laporan hasil analisis.

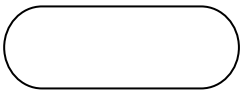
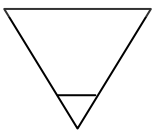
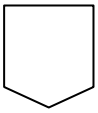
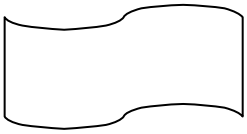
### 3.4.2 Desain Sistem

Desain sistem didefinisikan oleh Whitten (2004: 448) sebagai tugas yang fokus pada spesifikasi solusi detail berbasis komputer. Menurut Sutabri (2003: 88) tahap perancangan sistem ini merupakan prosedur untuk mengkonversi spesifikasi logis kedalam sebuah desain yang dapat diimplementasikan pada sistem komputer organisasi. Hasil akhir dari rancangan sistem ini adalah suatu laporan spesifikasi teknis dari bentuk-bentuk masukan dan keluaran serta spesifikasi teknis perangkat lunak yang akan berfungsi sebagai sarana pengolah data dan sekaligus penyaji informasi yang dibutuhkan.

### 3.4.3 *System Flow*

*System flow* merupakan alat bantu yang banyak digunakan untuk menggambarkan sistem secara pisikal (Tavri D. Mahyuzir, 1997: 41). Terdapat beberapa simbol yang digunakan untuk merancang sebuah desain dari sistem, simbol-simbol *system flow* tersebut dapat dilihat pada tabel 3.1 berikut ini.

Tabel 3.1 Simbol-Simbol *System Flow*.

| Simbol | Keterangan/ Fungsi |
|---|---|
|  | **Terminator**<br>Sebagai tanda dimulainya jalan proses sistem ataupun tanda akhir dari sebuah pengerjaan suatu sistem. |
|  | **Manual Operation**<br>Menggambarkan sebuah proses kerja manual. |
|  | **Document**<br>*Document* merupakan simbol dari dokumen yang berupa kertas laporan, surat-surat, memo, maupun arsip-arsip secara fisik. |
|  | **Process**<br>Merupakan sebuah bentuk kerja sistem yang dilakukan secara terkomputerisasi. |
|  | ***Database***<br>*Database* digunakan sebagai media penyimpanan data yang bersifat komputerisasi. |
|  | **Decision**<br>Merupakan operator logikadigunakan sebagai penentu keputusan dari suatu permintaan atau proses dengan dua nilai, benar atau salah. |
|  | **Manual Input**<br>Melakukan proses *input* ke dalam *database* melalui *keyboard*. |
|  | **Off-line Storage**<br>Merupakan media penyimpanan dokumen secara manual (arsip). |
|  | **On-page Reference**<br>Merupakan simbol untuk menghubungkan bagan desain sebuah sistem apabila hubungan arus data yang ada letaknya terlalu jauh. |
|  | **Off-page Reference**<br>Simbol ini digunakan apabila arus data yang ada dilanjutkan ke halaman yang berbeda. |
|  | **Paper Tape**<br>Merupakan simbol yang umumnya menggantikan bentuk penggambaran jenis pembayaran yang digunakan (misal: uang) dalam transaksi yang ada pada sistem yang dirancang. |

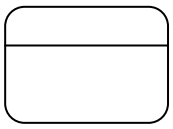
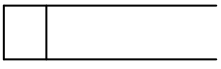
INSTITUT BISNIS & INFORMATIKA stikom SURABAYA

### 3.4.4 *Data* Flow *Diagram*

Untuk membaca suatu *Data Flow Diagram* (DFD) harus memahami dulu elemen-elemen yang menyusun suatu DFD.

Melalui suatu teknik analisa data terstruktur yang disebut *Data Flow Diagram* penganalis sistem dapat mempresentasikan proses-proses data di dalam organisasi. Pendekatan aliran data menekankan logika yang mendasari sistem, dengan menggunakan kombinasi dari empat simbol, penganalis sistem dapat menciptakan suatu gambaran proses-proses yang bias menampilkan dokumentasi sistem yang solid (Kendall, 2010: 263). Simbol-simbol yang digunakan dalam mendeskripsikan DFD dapat dilihat pada tabel 3.2.

Tabel 3.2 Simbol-Simbol *Data Flow Diagram*.

| Simbol | Keterangan/ Fungsi | Elemen Data Flow Diagram |
|---|---|---|
|  | **Simbol Entitas *Eksternal*/** Menggambarkan asal atau tujuan data. | Setiap entitas ekternal memiliki: nama dan deskripsi. |
|  | **Simbol Persegi/ Lingkaran** Menggambarkan entitas atau proses dimana aliran data masuk ditransformasikan ke aliran data keluar. | Setiap proses memiliki: nomor, nama, deskripsi proses, satu atau lebih output data flow, satu atau lebih input data flow. |
|  | **Simbol File/ *Data Store*** Menggambarkan tempat aliran data disimpan, | Setiap data store memiliki: nomor, nama, deskripsi, satu atau lebih output data flow, satu atau lebih input data flow. |

INSTITUT BISNIS & INFORMATIKA stikom SURABAYA

| Simbol | Keterangan/ Fungsi | Elemen Data Flow Diagram |
|---|---|---|
|  | **Simbol Aliran Data/ *Data Flow*** Menggambarkan aliran data. | Setiap data flow memiliki: nama, deskripsi, satu/lebih koneksi ke suatu proses. |

### 3.4.5 *Entity Relationship Diagram*

*Entity Relationship Diagram* (ERD) adalah gambaran pada sistem dimana di dalamnya terdapat hubungan antara *entity* beserta relasinya. Entity merupakan sesuatu yang ada dan terdefinisikan didalam suatu organisasi, dapat abstrak dan nyata. Untuk setiap entity biasanya mempunyai *attribute* yang merupakan ciri entity tersebut. Relasi adalah hubungan antar *entity* yang berfungsi sebagai hubungan yang mewujudkan pemetaan antar entity.

Leman (1998: 28) menyatakan bahwa ERD adalah diagram yang berfungsi untuk menggambarkan sistem yang terdiri dari hubungan entitas. Untuk menggambarkan sebuah ERD digunakan beberapa simbol. Pada dasarnya ada 3 (tiga) simbol yang digunakan, yaitu:

*a. Entity*

*Entity* merupakan objek yang mewakili sesuatu yang nyata dan dapat dibedakan dari sesuatu yang lain. Simbol dari entity*i*ni biasanya digambarkan dengan persegi panjang.

b. Atribut

Setiap entitas pasti mempunyai elemen yang disebut *atribut* yang berfungsi untuk mendeskripsikan karakteristik dari entitas tersebut. Isi dari atribut mempunyai sesuatu yang dapat mengidentifikasikan isi elemen satu dengan yang lain. Gambar *atribut* diwakili oleh simbol *elips*.

c. Hubungan/ Relasi

Hubungan antara sejumlah entitas yang berasal dari himpunan entitas yang berbeda. Relasi dapat digambarkan sebagai berikut :

1. *One to One*

   Hubungan relasi satu ke satu yaitu setiap entitas pada himpunan entitas A berhubungan paling banyak dengan satu entitas pada himpunan entitas B.

2. *One to Many*

   Setiap entitas pada himpunan entitas A dapat berhubungan dengan banyak entitas pada himpunan entitas B, tetapi setiap entitas pada entitas B dapat berhubungan dengan satu entitas pada himpunan entitas A.

3. *Many to Many*

   Setiap entitas pada himpunan entitas A dapat berhubungan dengan banyak entitas pada himpunan entitas B. Begitu juga pada entitas B, dapat berhubungan dengan banyak entitas A.

ERD ini diperlukan agar dapat menggambarkan hubungan antar entity dengan jelas, dapat menggambarkan batasan jumlah entity dan partisipasi antar entity, mudah dimengerti pemakai dan mudah disajikan oleh perancang *database*. Untuk itu ERD dibagi menjadi dua jenis model, yaitu:

a. *Conceptual Data Model* (CDM)

   *Conceptual Data Model* (CDM) adalah jenis model data yang menggambarkan hubungan antar tabel secara konseptual.

b. *Physical Data Model (PDM)*

   *Physical Data Model (PDM)* adalah jenis model data yang menggambarkan hubungan antar tabel secara fisikal.

INSTITUT BISNIS & INFORMATIKA stikom SURABAYA

## 3.5 Program yang Digunakan

### 3.5.1 Visual Basic

Visual Basic adalah salah suatu *development tools* untuk membangun aplikasi dalam lingkungan Windows. Visual Basic menggunakan pendekatan Visual untuk merancang *user interface* dalam bentuk *form*, sedangkan untuk kodingnya menggunakan dialek bahasa Basic yang cenderung mudah dipelajari.

Pada Pemrograman Visual, pengembangan aplikasi dimulai dengan pembentukan *user interface*, kemudian mengatur properti dari obyek-obyek yang digunakan dalam *user interface*, dan baru dilakukan penulisan kode program untuk menangani kejadian-kejadian (*event*).

### 3.5.2 *Database*

*Database* bisa dianggap sebagai kumpulan dari berbagai informasi yang diatur sedemikian rupa dan disimpan pada media tertentu agar informasi tersebut dapat dimanfaatkan/diambil kembali.

Ada tiga macam pembagian pada sistem *Database* :

1. Operasi kerja *database*
2. Bagian-bagian atau komponen sistem *database*
3. Konfigurasi sistem *database*

Pembagian berdasarkan operasi kerja meliputi :

1. Penyimpanan
2. Pengolahan
3. Pembacaan kembali

INSTITUT BISNIS & INFORMATIKA stikom SURABAYA

Penyimpanan digunakan untuk memasukkan data-data tertentu ke dalam *database*. Penyimpanan ini bisa berarti memasukkan data baru atau mengubah data yang lama. Data yang disimpan ke dalam *database* bisa berupa data mentah (data-data yang belum mengalami pengolahan), atau data-data yang siap digunakan (sudah mengalami pengolahan terlebih dahulu). Bentuk yang pertama memungkinkan proses penyimpanan menjadi sederhana, karena dari data asli langsung dimasukkan ke *database*. Selain itu, seluruh data asli masih kelihatan (tersedia) yang memungkinkan dilakukan pengolahan sesuai dengan kebutuhan. Bahkan jika kebutuhannya berubah, cukup dilakukan perubahan pada prosesnya, bukan pada datanya. Namun ini membawa konsekuensi, setiap kali diinginkan hasil akhirnya, harus selalu dilakukan pengolahan data.

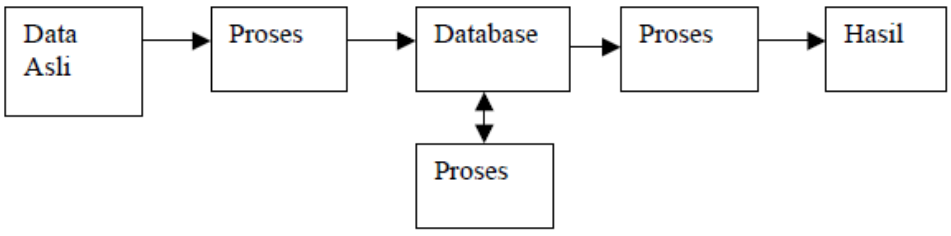
Bentuk yang kedua adalah dari data aslinya, sebelum dilakukan penyimpanan ke *database*, data asli tersebut diolah terlebih dahulu untuk mendapatkan hasil akhir. Setelah itu baru disimpan ke *database*. Cara ini mempercepat proses secara keseluruhan, karena pada saat memasukkan data, meskipun dilakukan pengolahan awal, datanya biasanya sedikit-sedikit sehingga tidak terasa. Sedangkan saat membaca hasil akhirnya, cukup dilakukan pembacaan langsung dari tabel yang jelas mempercepat proses. Namun cara ini akan banyak menghilangkan data-data aslinya, dan setiap terjadi perubahan sistem, akan banyak mengalami kesulitan.

Untuk penerapan yang sesungguhnya, dapat dipertimbangkan untuk menggabungkan antara kedua cara tersebut. Misalkan jika ada data aslinya yang memang suatu saat tidak akan pernah digunakan lagi, dapat langsung diolah. Kemudian jika dianggap proses yang akan dilakukan tidak memerlukan waktu

yang cukup panjang, sebaiknya data disimpan apa adanya, dengan proses dilakukan di belakang. Bisa juga digunakan data perantara, artinya dari data asli diolah sedikit untuk mendapatkan data pertengahan baru disimpan ke *database*. Ini akan sangat menghemat waktu.

Bagian Pengolahan digunakan untuk mendapatkan suatu informasi sesuai dengan yang diinginkan dari data aslinya. Proses pengolahan ini bisa dilakukan di awal sebelum data disimpan, bisa pada saat membaca hasil akhir, dapat juga dari data yang sudah disimpan di dalam *database* diambil kembali untuk diolah kemudian disimpan kembali.

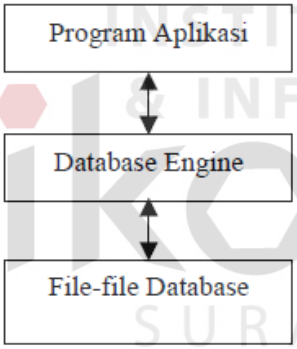
Pembacaan kembali digunakan untuk mengambil data yang ada di dalam data base untuk diolah, kemudian disimpan kembali atau dibuat laporannya. Ada beberapa hal yang bisa dilakukan pada saat pengambilan, antara lain mengambil seluruh isi dari sebuah tabel *database*, atau gabungan dari beberapa tabel *database*. Ini biasanya untuk ditampilkan dalam bentuk tabel. Bisa juga mengambil sebagian dari isi tabel (dengan difilter/diambil dengan kriteria tertentu), atau hanya mengambil satu baris data saja. Semua itu tergantung dari proses atau hasil yang ingin dilakukan.




Gambar 3.1 Proses dalam sistem *database*

### 3.5.3 Konfigurasi Sistem *Database*

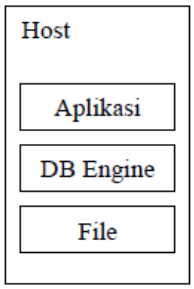
Suatu program aplikasi, saat melakukan operasi dengan *database*, harus melewati/menggunakan *database engine*. Perintah-perintah dari program aplikasi seperti membuat tabel, membuka tabel, membaca isi tabel dan sebagainya. Perintah-perintah ini diteruskan ke *database engine* untuk diolah lebih lanjut. *Database* engine bertanggung jawab untuk melakukan operasi pada file *database* secara fisik, misalkan suatu tabel/isi tabel disimpan pada disk yang mana dan di mana lokasinya. Suatu data pada saat disimpan ke disk, format datanya seperti apa. Dan seterusnya. Bahasa yang digunakan dari program aplikasi ke *database engine* hampir sama untuk setiap jenis *database*, terutama jika menggunakan perintah SQL. Sedangkan *database engine* harus selalu sesuai untuk setiap jenis *database* yang digunakan.




Gambar 3.2 Komponen sistem *database*

Konfigurasi dari sistem *database* meliputi

1. *Database* lokal
2. *Database* file server
3. *Database* client server




Gambar 3.3 *Database* lokal

Jika file *database*, *database engine* dan program aplikasi terletak pada komputer yang sama, maka konfigurasi ini disebut dengan *database* lokal. Jika program aplikasi dan *database engine* terletak pada satu komputer, tetapi file database terletak pada komputer lain dan terhubung dalam jaringan, maka konfigurasi ini disebut dengan *database* file server. Sedangkan jika file *database* dan *database engine* terletak pada satu komputer dan program aplikasi terletak pada komputer lainnya, maka ini disebut dengan *database client server*.

Konfigurasi *database* lokal memiliki konfigurasi yang paling sederhana, paling murah dan paling cepat. Tetapi jelas, jenis ini tidak dapat digunakan dalam mode *multi-user* atau dalam suatu jaringan.


Gambar 3.4 *Database* file server

*Database* file server memperbaiki fungsi *database* lokal dengan memungkinkan adanya mode *multi-user* atau kerja dalam suatu jaringan. Selain

itu, server yang digunakan tidak terlalu berat, karena hanya digunakan untuk menampung file *database* saja. Karena *database engine* terpisah jauh dengan file *database*, menyebabkan lalu-lintas jaringan menjadi relatif padat dan ujung-ujungnya sistem menjadi cukup lambat.


Gambar 3.5 *Database client server*

*Database client server* memperbaiki keduanya, yaitu dengan menawarkan mode *multi-user*, kerja jaringan dan akses yang cepat. Berhubung server harus menyimpan file *database* dan *database engine*, maka kerja *server* menjadi sangat berat. Karena itu, kinerja sistem *database client server* sangat ditentukan oleh kemampuan dari komputer server dan oleh kompleksitas dari *database engine*.

Dilihat dari cara penyimpanan/representasi data dalam *database*, dapat dibagi menjadi dua struktur *database*, *database* berbasis relasi (*relational database*) dan *database* berbasis obyek (*object oriented database*). Khusus untuk Oracle, struktur yang digunakan adalah berbasis relasi dan juga obyek. Artinya, Oracle dapat dipandang sebagai *database* relasi biasa, tetapi dapat juga dapat diperlakukan sebagai *database* obyek. Dalam hal ini, data-data yang disimpan ke dalam *database* Oracle bisa dalam bentuk object. Misalkan data-data object untuk penyimpanan multimedia.

Secara gampang, *database* relasi adalah *database* yang bekerja berdasarkan penyimpanan data dalam bentuk tabel dengan susunan baris dan kolom, dan antara satu tabel dengan tabel lainnya dapat saling dihubungkan/direlasikan. Sedangkan *database* yang berbasis obyek adalah, setiap data yang ada dianggap sebagai benda, sedangkan setiap benda sangat terkait dengan bendabenda lainnya yang terhubung seperti hirarki suatu keluarga (ada induk/orang tua/leluhur, dan ada anak/keturunan.

INSTITUT BISNIS & INFORMATIKA stikom SURABAYA