MODUL
BASIS DATA 1

Pengantar & Perancangan Database

Program Studi Bisnis Digital
Fakultas Ekonomi Universitas Garut
2022

ISBN 978-623-6918-13-5

Disusun Oleh :

- Diqy Fakhrun Shiddieq
- Fikri Fakhru Roji
- Dini Turipanan Alamanda

KATA PENGANTAR

Alhamdulillah puji syukur penulis panjatkan ke kahdirat Allah SWT, yang telah memberikan rahmat serta hidayahnya sehingga modul perkuliahan Basis Data ini dapat diselesaikan dengan baik. Dengan modul ini diharapkan dapat membantu mahasiswa dalam memahami dan mempelajari matakuliah Basis Data.

Penulis juga mengucapkan banyak terimakasih kepada semua pihak yang telah mendukung dalam penyusunan modul pembelajaran Basis Data ini baik berupa tenaga dan pikirannya. Terimakasih juga penulis sampaikan kepada rekan-rekan dosen, tenaga kependidikan di lingkungan Universitas Garut terutama Fakultas Ekonomi Program Studi Bisnis Digital dan semua pihak yang tidak dapat penulis sebutkan satu persatu.

Penulis menyadari dalam penyusunan modul pembelajaran Basis Data ini masih banyak kekurangan, untuk itu saran dan kritik yang membangun sangat penulis harapkan guna untuk perbaikan modul ini.

Akhir kata penulis berharap modul pembelajaran Basis Data ini dapat bermanfaat dan papat digunakan sebaik-baiknya untuk menjadi referensi dalam belajar Basis Data.

Garut, Februari 2022

Penulis,

DAFTAR ISI

KATA PENGANTAR i
DAFTAR ISI ii
BAB I KONSEP DASAR BASIS DATA 1
1.1. Pengenalan Basis Data 1
1.2. Pengertian Basis Data 2
1.3. Prinsip dan Tujuan Basis Data 3
BAB II SISTEM BASIS DATA 6
3.1. Komponen Sistem Basis Data 7
3.2. Komponen Sistem Basis Data 8
3.3. Bahasa Basis Data 11
BAB III BASIS DATA RELATIONAL 12
4.1. Definisi Basis Data Relational 12
4.2. *Constraint* / Batasan Basis Data Relational 14
4.3. Perancangan Basis Data 16
BAB IV MODEL DATA 18
4.1. Definisi Model Data 18
4.2. Jenis-Jenis Model Data 18
BAB V ENTITY RELATIONSHIP DIAGRAM (ER-D) 26
5.1. Definisi Entity Relationship Diagram (ER-D) 26
5.2. Simbol-Simbol ER-D 26
5.3. Tahapan Membuat ER-D 35
5.4. Logical Record Structured (LRS) 36
BAB VI TEKNIK NORMALISASI 39
6.1. Pengertian Normalisasi 39
6.2. Tujuan Normalisasi 40
6.3. Anomaly 41
6.4. Atribut dan Ketergantungan Fungsi 42
6.5. Tahapan Normalisasi 45
BAB VII BAHASA QUERY FORMAL 49
7.1. Pengantar Bahasa Query Formal 49
7.2. Operator Aljabar Relasional 49
BAB VIII BAHASA QUERY TERAPAN 53

8.1. Structured Query Language (SQL) 53
8.2. Pengelompokan Perintah Structured Query Language (SQL) 53
DAFTAR PUSTAKA 58

BAB I
KONSEP DASAR BASIS DATA

1.1. Pengenalan Basis Data

Basis data (*database*) dapat dibayangkan sebagai sebuah lemari arsip. Jika kita memiliki sebuah lemari arsip dan bertugas untuk mengelolanya, maka kemungkinan besar kita akan melakukan hal-hal seperti : memberi map pada kumpulan arsip yang akan disimpan, menentukan kelompok arsip, memberi penomoran dengan pola tertentu yang nilainya unik pada setiap map lalu menempatkan arsip-arsip tersebut dengan urutan tertentu didalam lemari. Kalaupun hal-hal tersebut tidak seluruhnya dilakukan, paling tidak, semua lemari arsip menerapkan suatu aturan tertentu tentang bagaimana keseluruhan arsip-arsip tadi disusun. Yang paling sederhana, tentu menyusun arsip-arsip tadi sesuai dengan kedatangannya (kronologisnya) dan tanpa pengelompokan. Hampir tidak akan pernah kita jumpai adanya lemari arsip yang tidak memiliki aturan dalam penyusunan arsip-arsip didalamnya.

Gambar 1-1 : Lemari Arsip

Ilustrasi lain dalam kehidupan sehari-hari misalnya, jika kita menyimpan pakaian kedalam lemari pakaian, maka hal paling sederhana yang akan dilakukan adalah : mengelompokan pakaian berdasarkan jenis pakaian. Kalaupun lemari pakaian dipakai bersama-sama, maka kita akan melakukan pengelompokan pakaian berdasarkan pemilik dan jenis pakaian.

Mengapa hal itu dilakukan? Jawabannya sederhana : kita berharap agar pada suatu saat nanti, sewaktu kita bermaksud untuk mencari dan mengambil kembali arsip atau buku atau baju dari lemari masing-masing, kita dapat melakukannya dengan mudah dan cepat (Fathansyah, 2015).

1.2. Pengertian Basis Data

Menurut terminologi bahasanya basis data terdiri dari dua kata, yaitu Basis dan Data. Basis adalah tempat, ruang atau markas berkumpul. Sedangkan Data adalah fakta yang menggambarkan atau perwujudan keadaan yang sebenarnya yang direkan dalam bentuk angka, teks atau yang lainnya.

Pengertian basis data menurut (Indrajani, 2015) adalah sebuah kumpulan data yang saling berhubungan secara logis, dan didesain untuk menemukan data yang dibutuhkan oleh sebuah organisasi.

Sedangkan menurut (Fathansyah, 2015) basis data dapat didefinisikan dalam sejumlah sudut pandang seperti :

a. Himpunan kelompok data (arsip) yang saling berhubungan yang diorganisasi sedemikian rupa agar kelak dapat dimanfaatkan kembali dengan cepat dan mudah.

b. Kumpulan data yang saling berhubungan yang disimpan secara bersama sedemikian rupa dan tanpa perulangan (redundansi) yang tidak perlu, untuk memenuhi berbagai kebutuhan.

c. Kumpulan *file*/tabel/arsip yang saling berhubungan yang disimpan dalam media penyimpanan elektronis.

Dari beberapa pengertian yang dikemukakan oleh beberapa tokoh diatas, dapat ditarik kesimpilan bahwa basis data adalah Kumpulan data yang saling berhubungan yang disimpan sedemikian rupa dengan tujuan agar tidak ada redundansi, dapat dimanfaatkan kembali dengan mudah dan cepat dan disimpan dalam media penyimpanan elektronik.

1.3. Prinsip dan Tujuan Basis Data

Basis data dan lemari arsip sesungguhnya memiliki prinsip kerja dan tujuan yang sama. Prinsip utamanya adalah pengaturan data/arsip. Dan tujuan utamanya adalah kemudahan dan kecepatan dalam pengambilan kembali data/arsip. Perbedaanya hanya terletak pada media penyimpanan yang digunakan, jika lemari arsip menggunakan lemari dari besi atau kayu sebagai media penyimpanannya, maka basis data menggunakan menggunakan media penyimpanan elektronis seperti cakram magnetis (Fathansyah, 2015).

Secara lebih lengkap, pemanfaatan basis data dilakukan untuk memenuhi sejumlah tujuan seperti berikut ini (Fathansyah, 2015) :

- Kecepatan dan Kemudahan

  Pemanfaatan basis data memungkinkan kita untuk dapat menyimpan data atau melakukan perubahan/manipulasi data atau menampilkan kembali data tersebut dengan lebih cepat dan mudah.

- Efisiensi Ruang Penyimpanan

  Dengan basis data, efisiensi/optimalisasi penggunaan ruang penyimpanan dapat dilakukan, karena kita dapat melakukan penekanan jumlah redundansi data.

- Keakuratan

  Pemanfaatan pengkodean atau pembentukan relasi antar data bersama dengan penerapan aturan atau batasan *(constraint)* tipe data, domain data yang secara ketat dapat diterapkan dalam basis data sangat berguna untuk menekan ketidakakuratan dalam penyimpanan data.

- Ketersediaan

  Sebuah basis data dapat memiliki data yang disebar dibanyak lokasi geografis. Sebagai contoh misalnya data nasabah sebuah bank disimpan dipisah-pisah sesuai dengan keberadaan nasabah. Dengan pemanfaatan teknologi jaringan computer, data yang berada suatu cabang, dapat juga diakses dicabang yang lain.

- Kelengkapan

  Lengkap atau tidaknya data yang kita kelola dalam sebuah basis data bersifat relatif. Seorang pemakai mungkin sudah menganggap bahwa data yang dikelola sudah lengkap, tetapi pemakai yang lain belum tentu berpendapat yang sama. Atau data yang dianggap sekarang sudah lengkap, belum tentu dimasa yang akan datang juga sama. Untuk mengakomodasi kebutuhan kelengkapan data yang semakin berkembang, kita tidak hanya dapat menambah data tetapi kita juga dapat melakukan perubahan terhadap struktur basis data.

- Keamanan

  Basis data memungkinkan kita untuk menerapkan aturan keamanan yang ketat, seperti menentuan siapa-siapa yang berhak menggunakan basis data dan menentukan operasi apa saja yang dapat dilakukannya.

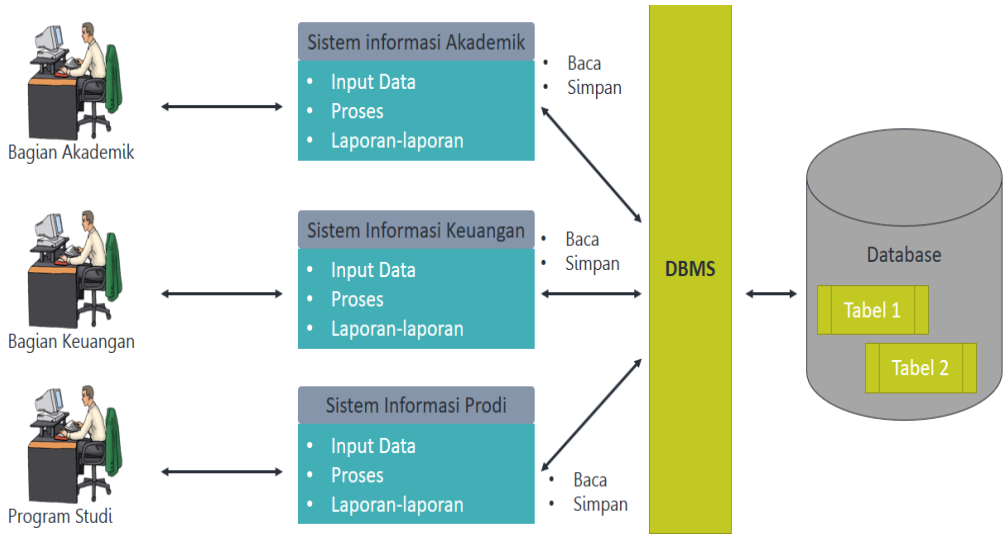
- Kebersamaan Pemakaian

  Pemakai basis data seringkali tidak terbatas pada satu pemakai saja, atau disatu lokasi saja atau oleh satu sistem/aplikasi saja. Data pegawai dalam basis data kepegawaian, misalnya, dapat digunakan oleh banyak pemakai dari sejumlah departemen dalam perusahaan atau oleh banyak sistem.

BAB II
SISTEM BASIS DATA

Sistem adalah sebuah tatanan (keterpduan) yang terdiri atas sejumlah komponen fungsional yang saling berhubungan dan secara bersama-sama bertujuan untuk memenuhi suatu proses tertentu. Sistem basis data merupakan sistem yang terdiri atas kumpulan tabel yang saling berhubungan (dalam sebuah basis data di sebuah sistem computer) dan sekumpulan program (yang biasa disebut DBMS) yang memungkinkan beberapa pemakai dan/atau program lain untuk mengakses dan memanipulasi tabel-tabel data tersebut (Fathansyah, 2015).

Sedangkan menurut (Connolly & Begg, 2015) sistem basis data adalah kumpulan dari program aplikasi yang berinteraksi dengan basis data bersama dengan *Database Management Systems* (DBMS) dan basis data itu sendiri.



Gambar 2-1: Sistem Basis Data

3.1. Komponen Sistem Basis Data

Dalam sebuah sistem basis data, terdapat enam (6) komponen-komponen utama atau pokok yang dijabarkan sebagai berikut :

1. Perangkat Keras (*Hardware*)

   Perangkat keras yang terdapat dalam sistem basis data adalah :

   - Komputer (satu untuk sistem yang *stand-alone* atau lebih dari satu untuk sistem jaringan)
   - Memori sekunder *on-line* (*Harddisk*)
   - Memori sekunder *off-line* (*Tape* atau *Removable Disk*) untuk keperluan *backup* data
   - Media/perangkat komunikasi (untuk sistem jaringan)

2. Sistem Operasi (*Operating Systems*)

   Sistem operasi adalah program yang mengaktifkan sistem komputer, mengendalikan seluruh sumber daya dalam komputer dan melakukan operasi-operasi dasar dalam komputer. Contoh sistem operasi yang banyak digunakan adalah : MS-DOS, MS-Windows, Linux dan Unix.

3. Basis Data (*Database*)

   Dalam sebuah sistem basis data dapat memiliki lebih dari satu basis data. Setiap basis data terdiri dari sejumlah objek basis data seperti : tabel, *index*, *procedure*, *function*, *trigger* dal lain-lain.

4. Sistem Pengolahan Basis Data (DBMS)

   Perangkat lunak yang memungkinkan pemakai untuk mengelola basis data disebut dengan Database *Management Systems* (DBMS). Perangkat lunak

yang termasuk kedalam DBMS adalah : Ms. SQL Server, Oracle Database, IBM DB2, MySQL, PostgreSQL.

5. Pemakai (*User*)

Ada beberapa jenis pengguna basis data yang berinterakasi langsung dengan basis data melalui perangkat lunak DBMS, diantaranya :

- Database Administrator (DBA) adalah pengguna yang membuat dan mengontrol akses kedalam basis data.
- Programmer adalah pengguna yang membuat aplikasi basis data yang digunakan oleh DBA dan pengguna akhir.
- Pemakai Akhir adalah pengguna yang melakukan manipulasi data (menambah, menghapus, mengubah dan mengakses data).

6. Aplikasi (Perangkat Lunak) lain

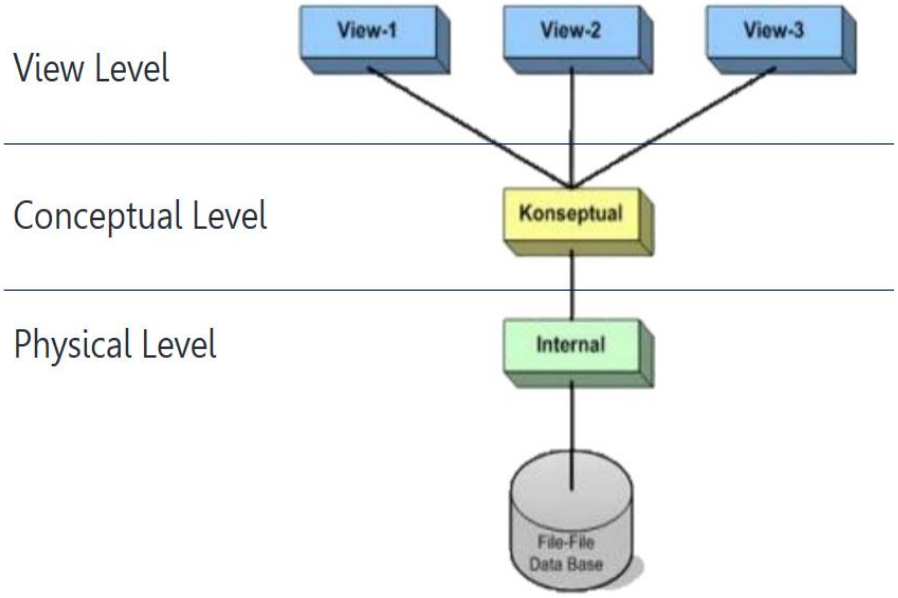
Aplikasi lain bersifat optional tergantung pada kebutuhan kita. Aplikasi ini biasanya digunakan untuk membantu pengguna dalam melakukan pengisian, perubahan dan pengambilan data. Program ini ada yang sudah disediakan bersama dengan DBMS-nya namun ada juga yang dibuat sendiri menggunakan aplikasi lain yang khusus.

3.2. Komponen Sistem Basis Data

Salah satu tujuan dari DBMS adalah untuk menyediakan fasilitas/antarmuka (*interface*) kepada user, untuk itu system tersebut akan menyembunyikan detail tentang bagaimana data disimpan dan dipelihara, sehingga data yang terlihat oleh user sebenarnya berbeda dengan yang tersimpan secara fisik.

Ada 3 tingkat dalam arsitektur basis data yang bertujuan membedakan cara pandang pemakai (*user*) terhadap basis data :

- *View Level*
- *Conceptual Level*
- *Physical Level*



Gambar 2-2: Abstaksi Basis Data

1. *Physical Level*

Level terendah dalam abstraksi data, yang menunjukkan bagaimana sesungguhnya suatu data disimpan. Tingkatan ini berurusan dengan :

- Alokasi ruang penyimpanan untuk data dan *indeks*
- Deskripsi *record* untuk penyimpanan
- Penempatan *record* data
- Teknik kompresi dan enkripsi data

Level ini paling komplek, struktur data level terendah. Menggambarkan bagaimana data disimpan dalam kondisi sebenarnya.

2. *Conceptual Level*

Level ini menggambarkan data apa (*what*) yang sebenarnya disimpan dalam basis data dan hubungannya dengan data yang lain. Hal-hal yang digambarkam dalam *level conceptual* adalah :

- Semua entitas beserta atribut dan hubungannya
- Batasan data
- Informasi semantik tentang data
- Keamanan dan integritas informasi.

3. *View Level*

Level ini merupakan level tertinggi dalam abstraksi basis data yang hanya menunjukan sebagian dari basis data. Hal-hal yang digambarkam dalam *level View* adalah :

- Hanya sebagian saja yang dilihat dan dipakai,disebabkan pemakai *database* tidak memakai semua isi *database*.
- Level yang sangat dekat dengan *user*/pemakai.
- Setiap user butuh sebagian dari dari *database*

Sebagai contoh : Bagian personalia rumah sakit hanya memakai data *file* pegawai dan gaji, tidak membutuhkan *file* pasien, resep dan pendaftaran.

3.3. Bahasa Basis Data

Merupakan bahasa yang digunakan oleh Pengguna untuk berkomunikasi / berinteraksi dengan DBMS yang bersangkutan. Bahasa basis data terbagi atas dua jenis, yaitu :

1. *Data Definition Languange* (DDL)

   Bahasa yang digunakan untuk melakukan mendefinisikan (membuat, menghapus, merubah) objek yang ada dalam basis data. Dengan bahasa ini kita dapat membuat tabel baru, membuat indeks, mengubah tabel, menentukan struktur penyimpanan tabel dan lain-lain.

2. *Data Manipulation Languange* (DDL)

   Bahasa yang digunakan untuk melakukan manipulasi data yang ada dalam basis data. Adapun operasi yang dapat dilakukan adalah menyisipkan data, modifikasi data, memanggil data dan menghapus data.

BAB III
BASIS DATA RELATIONAL

4.1. Definisi Basis Data Relational

Basis data relational adalah basis data yang mempresentasikan data dalam bentuk tabel dua dimensi yang terdiri atas baris dan kolom untuk memberikan gambaran sebuah berkas data. Basis data akan disebar atau dipilah-pilah kedalam berbagai tabel dua dimensi, setiap tabel terdiri atas lajur mendatar yang disebut baris data (*record*/*row*) dan lajur vertical yang biasa disebut dengan kolom (*column*/*field*).

Untuk mengilustrasikan model data relational lebih nyata, berikut kita ambil contoh basis data sederhana yang berkaitan dengan sistem akademik, sebagai berikut :

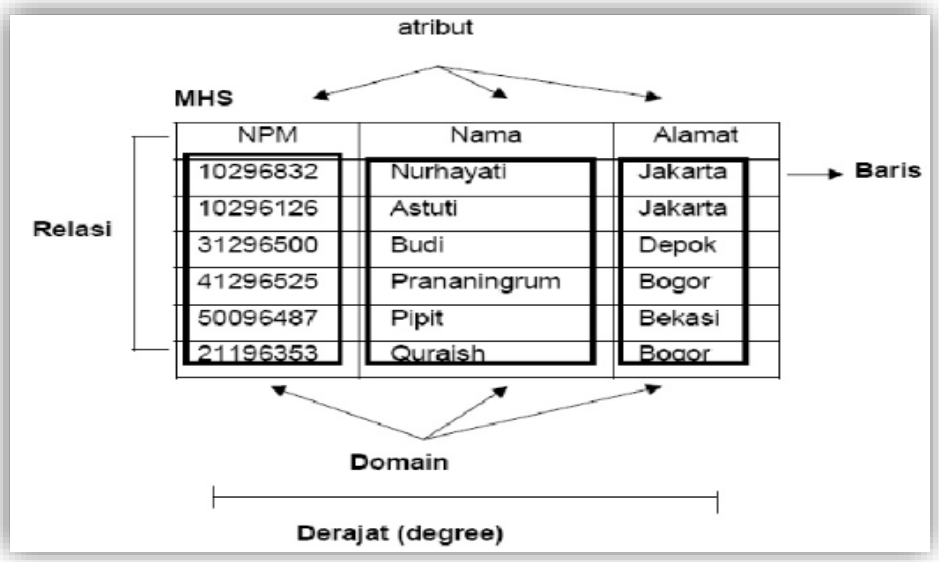
Data Mahasiswa

NPM	Nama	Alamat	Jenis Kelamin
101201	Bekham Putra	Jl. Pasir Kaliki Bandung	Laki-laki
101202	Teja Paku Alam	Jl. Ambon Bandung	Laki-laki
101203	Ahmad Jufriyanto	Jl. Ahmad Yani Bandung	Laki-laki

Dari data diatas kita dapat mengetahui karakteristik dari tabel, seperti berikut ini :

- Data Mahasiswa adalah nama dari tabel
- Data mahasiswa memiliki empat buah kolom data yaitu : NPM, Nama, Alamat dan Jenis Kelamin
- Data mahasiswa memiliki tiga baris data.

Untuk lebih detail mengenai istilah-istilah yang ada pada model basis data relational, akan dijelaskan berdasarkan Gambar berikut :



Gambar 3-1: Model Data Relational

- Relasi/tabel/*file*

  Representasi tabel yang terdiri atas sejumlah baris dan sejumlah kolom. Tabel pada sebuah basis data tidak boleh memiliki nama yang sama.

- *Tuple*/*row* /*record*

  Sebuah baris pada tabel, harus memiliki nilai yang unik dan ditempatkan dalam urutan bebas karena tidak mempengaruhi makna dari tabel.

- *Attribute*/ *field*/*column*

  Suatu kolom pada table, harus memiliki nilai yang unik dan urutan kolom bebas karena tidak mempengaruhi makna dari tabel.

- *Domain*

  Sekumpulan nilai yang dapat disimpan dalam satu atau lebih *attribute*/*Field*. Sebuah domain dapat dimiliki oleh lebih dari satu kolom,

tetapi satu kolom hanya dapat memiliki satu domain. Karena domain dapat mengatur jenis nilai yang disimpan, maka disebut domain *constraint*.

- *Cardinality*

  Banyaknya tupel/baris/Record pada tabel

- *Degree*

  Banyaknya attribute atau kolom/Field pada tabel

4.2. *Constraint* / Batasan Basis Data Relational

Ada tiga jenis *constraint* atau batasan yang terdapat dalam model basis data relational yaitu : *Domain Constraint*, *Key Constraint* dan *Integrity Constraint*.

a. *Domain Constraint*

Menyatakan hahwa nilai data sesuai dengan kenyataannya. Sistem dalam hal ini melakukan *CHECK* sesuai dengan aturan (*rule*) yang berlaku. Batasan dan implementasi (*DEFAULT*, *CHECK*, *FOREIGNKEY*) Tipe data, *default*, *rules*. Contoh pada data mahasiswa diatas, kolom jenis kelamin hanya boleh diisi oleh dua nilai yaitu laki-laki atau perempuan.

b. *Key Constraint*

Suatu tabel adalah satu set record, dan semua elemen dalam satu set harus berbeda, tidak boleh terdapat dua record yang mempunyai nilai yang sama untuk semua atributnya. Untuk menerapkan aturan seperti itu dengan menggunakan *key constraint*. Beberapa jenis *key constraint* yang terdapat dalam model data relational diantaranya :

1) *Superkey*

Sebuah *attribute*, atau kumpulan *attribute* yang dapat mengidentifikasi tuple dalam sebuah relasi

2) *Candidate key*

Merupakan *superkey* minimal, sehingga tidak ada subset dari *superkey* ini yang dapat mengidentifikasi *tuple* dalam sebuah relasi

3) *Primary key*

Candidate key yang dipilih untuk mengidentifikasi *tuple*/*Record* secara unik dalam sebuah relasi/Tabel (Ditandai dengan garis bawah) / *icon* kunci

4) *Foreign Key*

Satu atau lebih *attribute*/*Field* dalam sebuah relasi / tabel yang sama dengan *primary key* dari relasi /tabel lain yang berhubungan dengannya.

5) *Alternate key*

Candidate key atau *field* yang tidak dipilih menjadi *primary key*

c. *Integrity Constraint*.

Jenis *constraint* lain yang berfungsi untuk menjaga integritas data dalam basis data relational. Terdapat dua jenis *integrity constraint* yang umum digunakan untuk menjaga interitas data :

1) *Entity integrity*

Adalah batasan atau aturan yang menyatakan bahwa kolom dengan *primary key* tidak boleh memiliki atau menyimpan nilai *null*.

2) *Referential intergity*

Adalah batasan atau aturan yang digunakan untuk melakukan relasi antar tabel (menghubungkan atribut data antar tabel).

4.3. Perancangan Basis Data

Tujuan dari perancangan basis data adalah :

1. Untuk memenuhi informasi yang berisi kebutuhan-kebutuhan pengguna secara khusus beserta aplikasinya.
2. Memudahkan pengertian struktur informasi
3. Mendukung kebutuhan-kebutuhan pemrosesan dan beberapa objek penampilan (*respone time*, *processing time* dan *storage space*).

Proses atau tahapan pembangunan basis data terdiri dari dua tahapan utama :

1. Tahap analisis dan perancangan

   Ada tiga proses yang dilakukan pada tahapan analisis dan perancangan :

   1) Perancangan basis data secara konsep

      Merupakan proses pembuatan data model atau skema konseptual dan perancangan transaksi yang akan terjadi pada basis data.

   2) Perancangan basis data secara logik

      Merupakan proses transformasi atau pemetaan data, transformasi ini tidak tergantung pada sistem.

   3) Perancangan basis data secara fisik

      Merupakan proses deskripsi implementasi basis data pada media penyimpanan sekunder (*disk*).

2. Tahap implementasi

Tahapan ini menerapkan rancangan basis data yang dibuat pada tahapan sebelumnya. Penerapan basis data ini menggunakan aplikasi yang disediakan oleh DBMS yang dipilih.

BAB IV
MODEL DATA

4.1. Definisi Model Data

Sekumpulan perangkat konseptual untuk mendeskripsikan atau menerangkan data, relasi-relasi / hubungan antar data yang menyatakan hubungan antar rekaman yang tersimpan dalam basis data serta batasan data.

4.2. Jenis-Jenis Model Data

Dalam basis data terdapat dua jenis model data yang umum digunakan yaitu: model data berbasis *object* dan model data berbasis *record*.

A. Model data berbasis *object*

Model data berbasis objek menggunakan konsep entitas, atribut dan hubungan antar entitas (relationship). Entitas itu sendiri adalah segala sesuatu apapun yang terdapat dalam sebuah sistem baik nyata maupun abstrak. Sedangkan atribut adalah property atau karakteristik yang dimiliki oleh sebuah entitas. Sedangkan relationship adalah hubungan yang terjadi antara dua entitas atau lebih. Model data berbasis objek terdiri dari :

1. Model Keterhubungan Entitas (Entity Relationship Model)

   a. Pengertian

   Model keterhubungan entitas merupakan model yang paling populer digunakan dalam perancangan data. Model ini digunakan untuk menjelaskan hubungan antar data dalam basis data berdasarkan suatu persepsi bahwa *real word* terdiri dari objek-objek dasar yang mempunyai hubungan atau relasi antar objek-objek tersebut. Komponen utama

pembentuk model ini yaitu: Entitas (*entity*) dan Relasi (*relationship*). Kedua komponen ini dideskripsikan lebih lanjut melalui sejumlah Atribut atau Properti.

b. Simbol

Simbol dasar yang digunakan dalam *Entity Relationship Model* adalah:

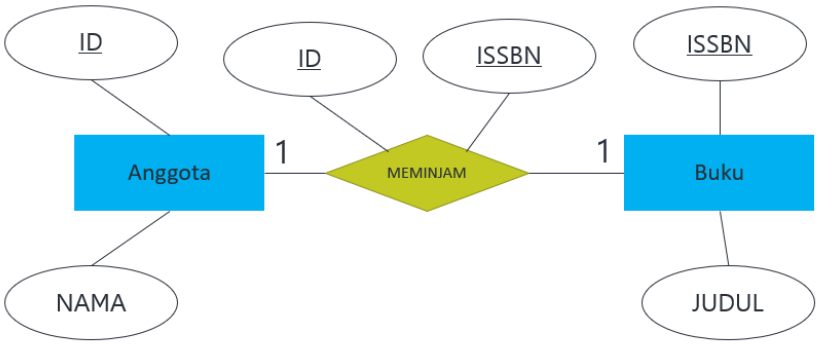
Tabel 4-1: Simbol dasar Entity Relationship Model

Simbol	Keterangan
	Menggambarkan satu set entitas
	Menggambarkan himpunan relasi
	Penghubung antara relasi entitas dan atribut
	Menggambarkan satu atribut

c. *Mapping Cardinality*

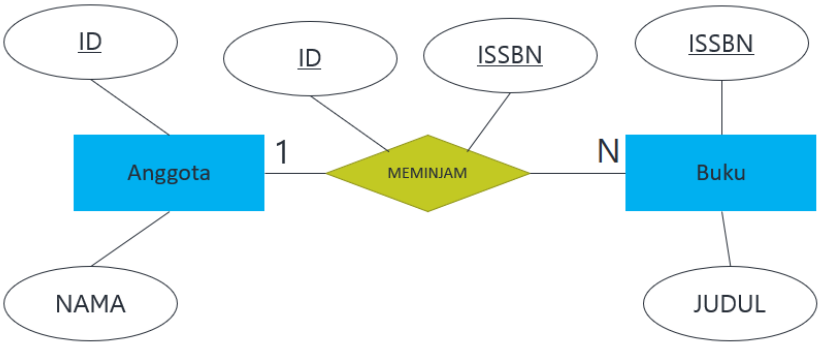
Dalam *Entity Relationship Model* aturan terpenting adalah kardinalitas relasi atau *mapping cardinality* yang menentukan jumlah entity yang dapat dihubungkan dengan entity lainnya melalui relationship-set. Jenis *Mapping Cardinality* :

(1) Relasi satu ke satu (*One to One*)



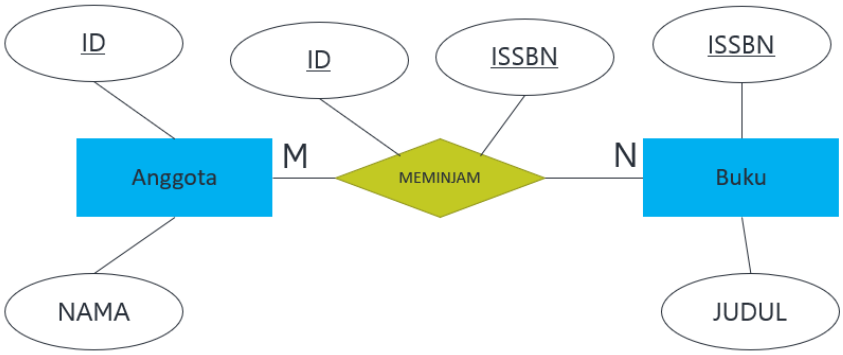
Gambar 4-1: Contoh Relasi satu ke satu

(2) Relasi satu ke banyak (*One to Many*)



Gambar 4-2: Contoh Relasi satu ke banyak

(3) Relasi banyak ke banyak (*Many to Many*)



Gambar 4-3: Contoh Relasi Banyak ke Banyak

2. Model Berorientasi Objek (Object Oriented Model)

Penggambaran model data berbasis objek menggunakan diagram *Unified Modeling Language* (UML). UML ini terdiri dari 2 jenis diagram yaitu : *Structural* diagram dan *Behaviour* diagram.

Structural diagram terdiri dari empat diagram yaitu : *Class Diagram, Object Diagram, Componen Diagram dan Deployment Diagram.* Sedangkan *behavior* Diagram terdiri dari lima diagram yaitu : *Use Case Diagram, Sequence Diagram, Communication Diagram, Statechart Diagram* dan *Activiy Diagram.*

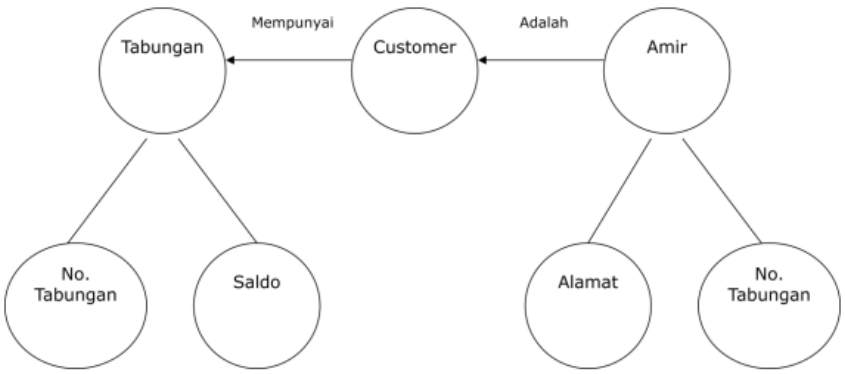
3. Model Data Semantik (Semantic Data Model)

Hampir sama dengan *Entity Relationship Model* dimana relasi antar objek-objek dasar tidak dinyatakan dengan simbol tetapi menggunakan kata-kata (semantik). Sebagai contoh, akan digambarkan hubungan data pada Bank X. Tanda-tanda yang digunakan dalam semantik model adalah :

——————→ : Menunjukan adanya relasi

———————— : Menunjukan adanya atribut

Contoh :



Gambar 4-4: Contoh Semantik Model

4. Model Data Fungsional (Functional Data Model)

Model yang terakhir dari tiga model pada teknik pemodelan objek. Fokus utama model fungsional adalah untuk mendapatkan logika program, hubungan antarproses objek-objek dan data yang terlibat dalam sistem

B. Model data berbasis *record*

Model ini berdasarkan pada *record* untuk menjelaskan kepada pengguna tentang hubungan *logic* antar data dalam basis data. Perbedaan dengan model data berbasis objek adalah pada model berbasis *record* disamping digunakan untuk menguraikan struktur logika keseluruhan suatu *database*, juga digunakan untuk menguraikan implementasi dari sistem *database*. Ada tiga jenis model data berbasis *record* yaitu :

1. Model Data Relational

Dimana data serta hubungan antar data dipresentasikan dalam sejumlah tabel dan masing-masing tabel terditi dari sejumlah kolom dan baris.

Contoh : Dalam database penjualan terdapat tiga buah tabel seperti Sipplier, suku cadang dan pengiriman.

No_supl	Nama_pen	Status	KOTA
S01	PT. OGAH-RUGI	03	MEDAN
s02	PT. SANTAI-DULU	03	SURABAYA
s03	PT. MALU-MALU	02	BANDUNG
s04	CV. ASAL JADI	01	MEDAN

NO_SUPL	NO_PART	JUML
S01	P01	200
S01	P02	300
S01	P04	250
S02	P01	300
S02	P02	400
S03	P03	400
S04	P04	300

PENGIRIMAN

SUKU CADANG

NO_PART	NAMA_PART	BAHAN BAKU	BERAT	KOTA
P01	BAUT-3 cm	BESI	10	JAKARTA
P02	MUR - 3 cm	BESI	8	JAKARTA
P03	BAUT -10 cm	ALUMUNIUM	45	SURABAYA
P04	PACKING	KARET	6	MEDAN
P05	RING-MM	ALUMUNIUM	2	JAKARTA

Gambar 4-5: Contoh Model Relational

Ada beberapa sifat yang melekat pada sebuah tabel :

1) Tidak boleh ada *record* yang sama (kembar)

2) Urutan *record* tidak terlalu penting, karena data dalam *record* dapat diurut sesuai dengan kebutuhan.

3) Setiap *field* harus mempunyai nama yang unik (tidak boleh ada yang sama).

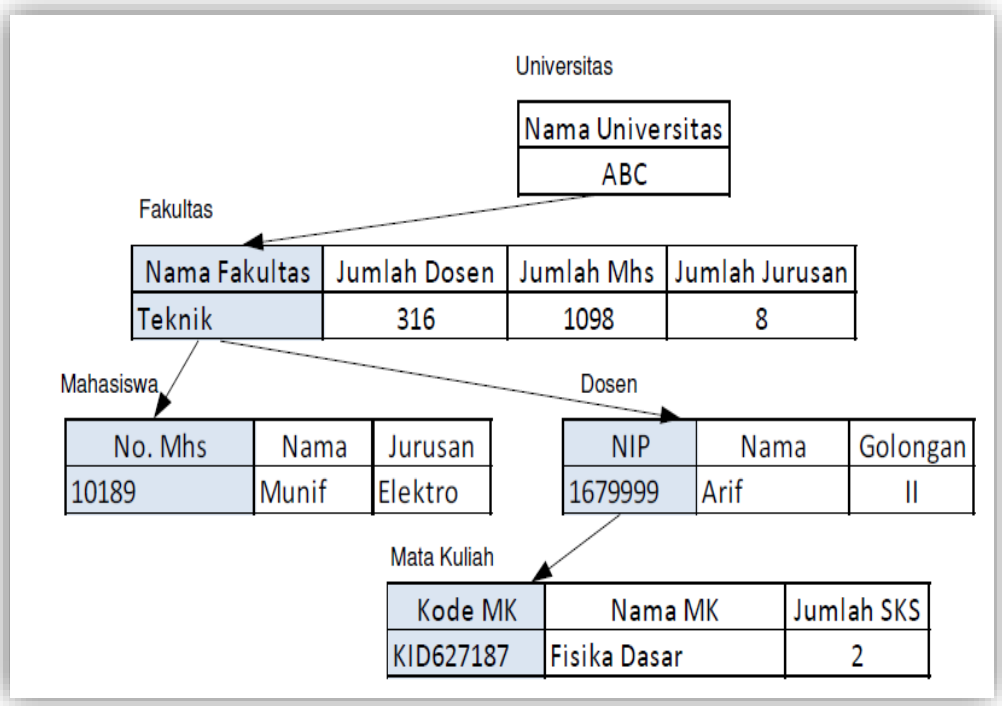
4) Setiap *field* mesti mempunyai tipe data dan karakteristik tertentu

2. Model Data Hirarki

Data serta hubungan antar data direpresentasikan dalam bentuk record dan link (pointer), dimana record-record disusun dalam bentuk pohon. Pada puncak hirarki disebut dengan akar (root). Tiap record tingkat atas (parent) mempunyai satu atau lebih sub-record (children) sehingga setiap entitas hanya boleh mempunyai satu induk, tetapi dapat

mempunyai banyak anak. Pada model data hirarki, hubungan antar record dinyatakan dalam satu-banyak (one to many) atau satu-satu (one to one).

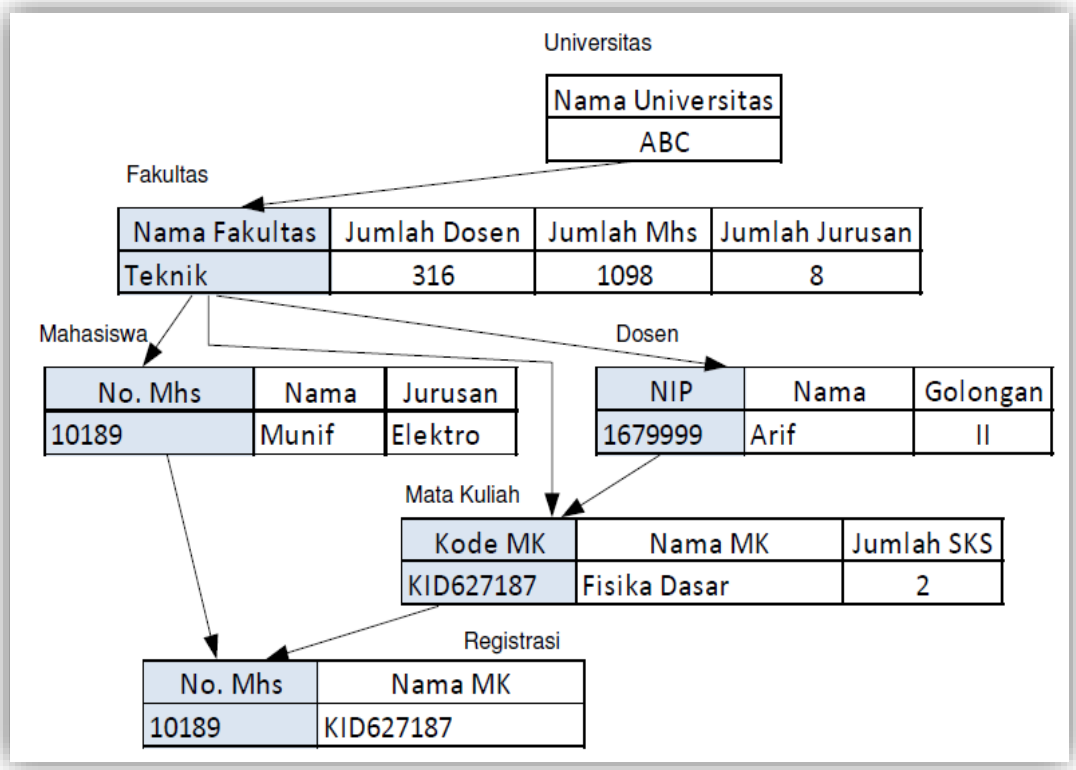
Contoh : Dalam satu Universitas terdapat banyak Fakultas dan setiap Fakultas terdapat banyak Dosen atau banyak Mahasiswa, dan seterusnya. Tanda panah menunjukkan derajat keterhubungan “banyak”.



Gambar 4-6: Contoh Model Hirarki

3. Model Data Jaringan

Model ini hampir sama dengan model hirarki. Perbedaannya dalam model ini setiap record dapat mempunyai banyak induk dan banyak anak. Dalam model ini lebih sedikit terdapat data rangkap, namun lebih banyak terdapat hubungan antar record, sehingga akan menambah informasi hubungan yang harus disimpan dalam database. hal ini akan menambah volume dan kerumitan dalam penyimpanan berkas data.



Gambar 4-7: Contoh Model Jaringan

BAB V
ENTITY RELATIONSHIP DIAGRAM (ER-D)

5.1. Definisi Entity Relationship Diagram (ER-D)

Sebuah teknik pemodelan data yang merepresentasikan gambar entitas dan relasi-relasi antar entitas di dalam sebuah sistem informasi. Menurut (Fathansyah, 2015) *Model Entity Relationship* berisi komponen-komponen himpunan entitas dan himpunan relasi yang masing-masing dilengkapi dengan atribut-atribut yang merepresentasikan seluruh fakta dari dunia nyata yang kita tinjau.

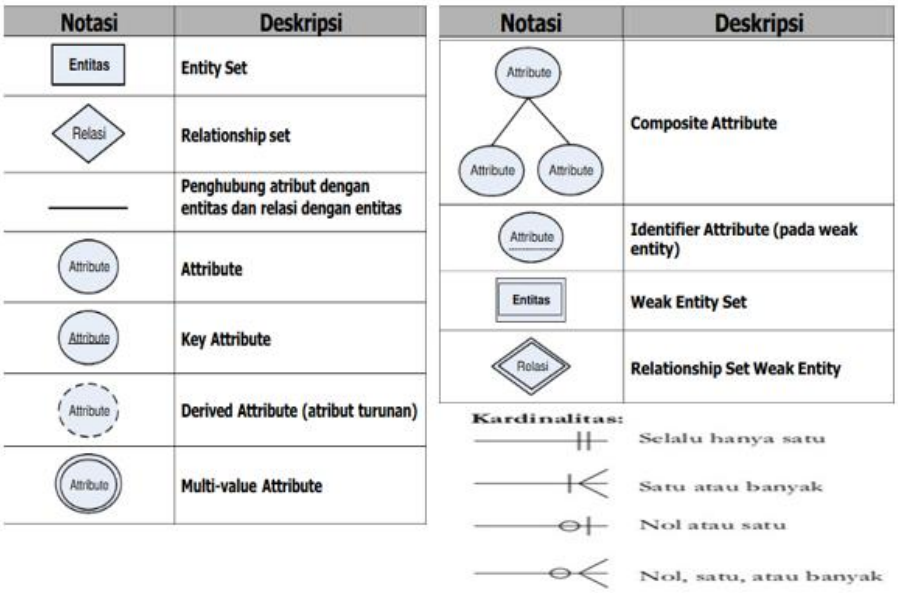
Entity Relationship Diagram dibentuk oleh tiga komponen utama yaitu :

a) Entitas (Entity)

b) Relasi (Relationship)

c) Atribut (Attribute)

5.2. Simbol-Simbol ER-D



Gambar 5-1: Simbol-Simbol ER-D

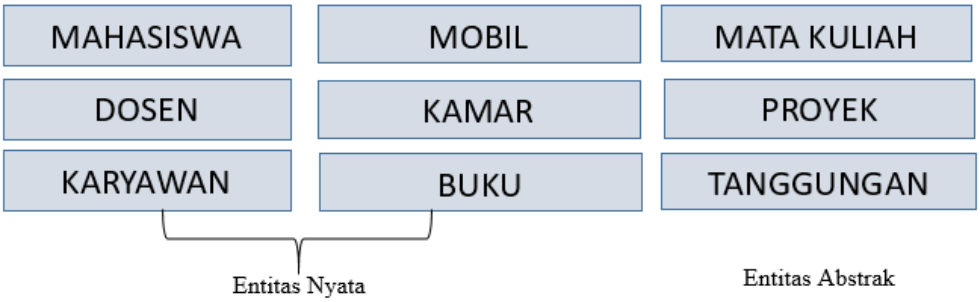
1. Entitas

Entity (entitas) adalah sesuatu apa saja (object) yang ada di dalam sistem, nyata maupun abstrak dimana data tersimpan atau di mana terdapat data.

Aturan menggambar entitas :

1) Entitas dinyatakan dengan simbol persegi panjang
2) Nama entitas berupa kata benda tunggal
3) Nama entitas mudah dipahami dan jelas

Berikut adalah contoh penggambaran entitas :

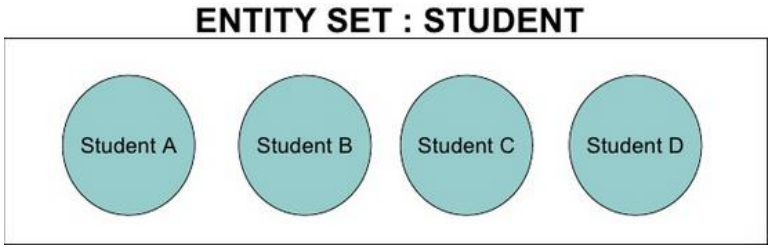


Gambar 5-2: Conoh Simbol Entitas

Entitas Set :

Entitas set adalah kumpulan atau gabungan dari beberapa entitas yang memliki karakteristik sejenis. Entitas set berupa :

1) Objek secara fisik : Rumah , Kendaraan, Peralatan
2) Objek secara konsep : Pekerjaan, Rencana, Aktivitas lain



Gambar 5-3 : Contoh Entitas Set

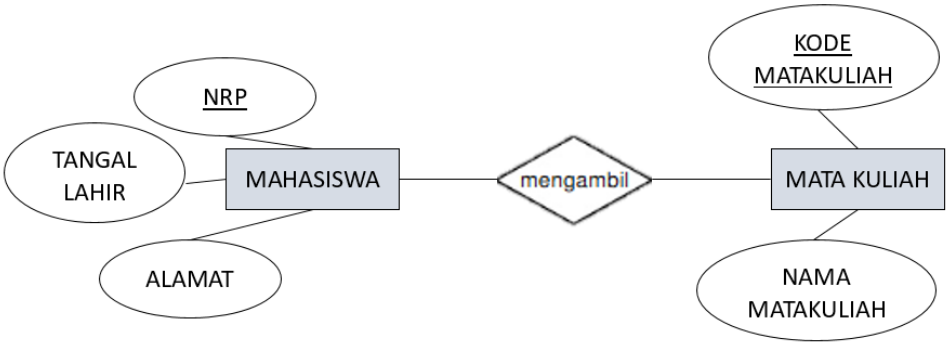
2. Relasi

Adalah hubungan/ kejadian / transaksi yang terjadi antara 2 buah entitas yang keterangannya perlu di simpan dalam basis data.

Aturan menggambar relasi:

1) Kerelasian dinyatakan dgn simbol belah ketupat
2) Menghubungkan 2 entitas
3) Mata kerja aktif dan tunggal

Berikut adalah contoh penggambaran relasi :



Gambar 5-4 : Contoh Relasi

2.1. Derajat Relasi

Derajat relasi bertujuan untuk menjelaskan jumlah entitas yang berpartisipasi dalam suatu relationship. Ada tiga jenis derajat relasi yaitu :

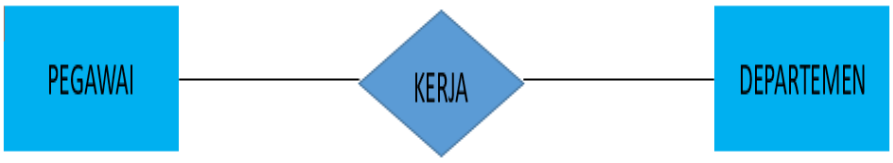
- Unary Degree(Derajat Satu)

  Sebuah relasi dalam ER-D yang menghubungkan satu entitas yang sama.

Gambar 5-5: Contoh Unary

- Binary Degree(Derajat Dua)

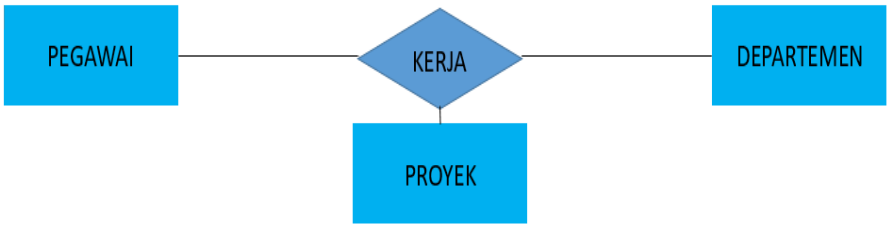
  Sebuah relasi dalam ER-D yang menghubungkan dua entitas yang berbeda.



Gambar 5-6: Contoh Binary

- Ternary Degree(Derajat Tiga)

  Sebuah relasi dalam ER-D yang menghubungkan tigas entitas yang berbeda.



Gambar 5-7: Contoh Ternary

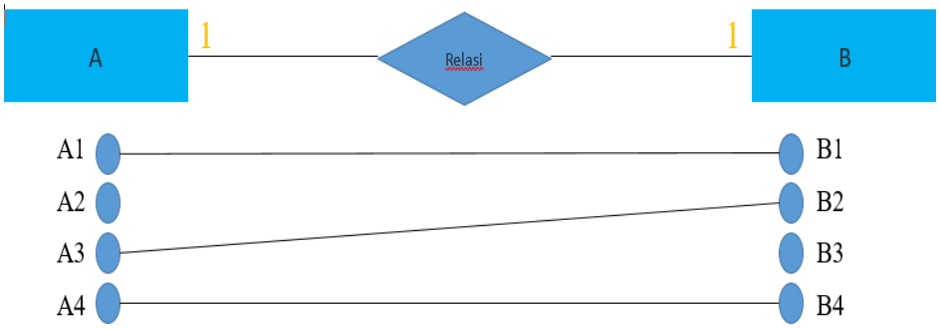
2.2. kardinalitas Relasi

Menjelaskan batasan jumlah keterhubungan satu entity dengan entity lainnya. Ada tiga kardinalitas relasi dalam ER-D yaitu :

- 1:1 (One to One)
- 1:N (One to Many)
- M:N (Many to Many)

A. 1:1 (One to One)

Terjadi bila setiap anggota dari entitas A hanya boleh berpasangan dengan satu anggota dari entitas B, dan sebaliknya.



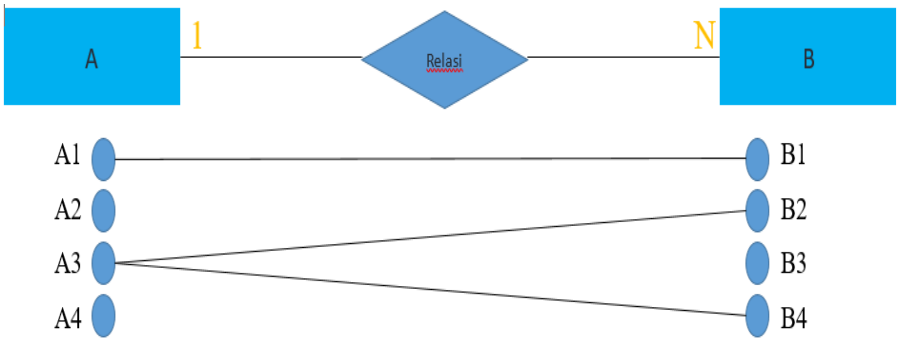
Gambar 5-8: Kardinalitas One To One

Contoh dalam kasus dalam sistem perusahaan, dimana satu orang pegawai memiliki satu kendaraan dan satu kendaraan hanya dimiliki oleh satu pegawai. Maka penggambaran ER-D nya adalah :

Gambar 5-9: Contoh One to One Kasus Perusahaan

B. 1:N (One to Many)

Terjadi bila setiap anggota dari entitas A boleh berpasangan dengan lebih dari satu anggota dari entitas B. Sebaliknya entitas B hanya boleh berpasangan dengan satu anggota dari entitas A.



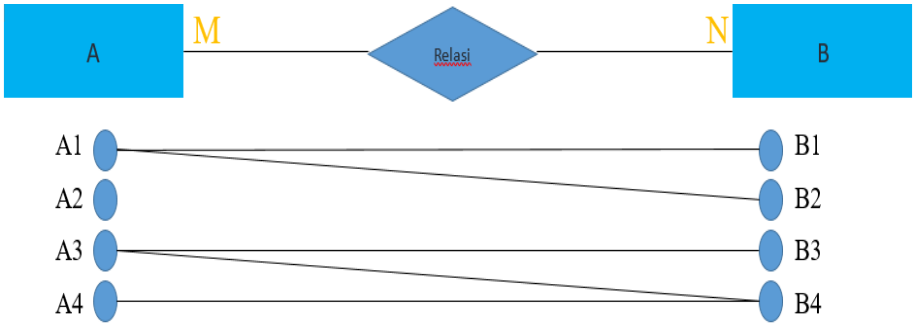
Gambar 5-10: Kardinalitas One To Many

Contoh dalam kasus dalam sistem perkuliahan, dimana satu orang mahasiswa dapat mengambil satu atau lebih dari satu matakuliah, sedangkan satu matakuliah hanya bisa diikuti oleh satu orang mahasiswa. Maka penggambaran ER-D nya adalah :

Gambar 5-11: Contoh One To Many pada Kasus Perkuliahan

C. M:N (Many to Many)

Terjadi bila setiap anggota dari entitas A boleh berpasangan dengan lebih dari satu anggota dari entitas B. dan Sebaliknya.



Gambar 5-12: Kardinalitas Many To Many

Contoh dalam kasus dalam sistem perusahaan, dimana satu orang pegawai dapat terlibat dalam satu atau lebih dari satu proyek, dan satu proyek dapat dikerjakan oleh satu atau lebih dari satu pegawai. Maka penggambaran ER-D nya adalah :

Gambar 5-13: Contoh Many To Many Pada Kasus Perusahaan

3. Atribut

Ciri dan karaktereistik dari suatu entitas. Biasanya direpresentasikan sebagai kolom data dari field

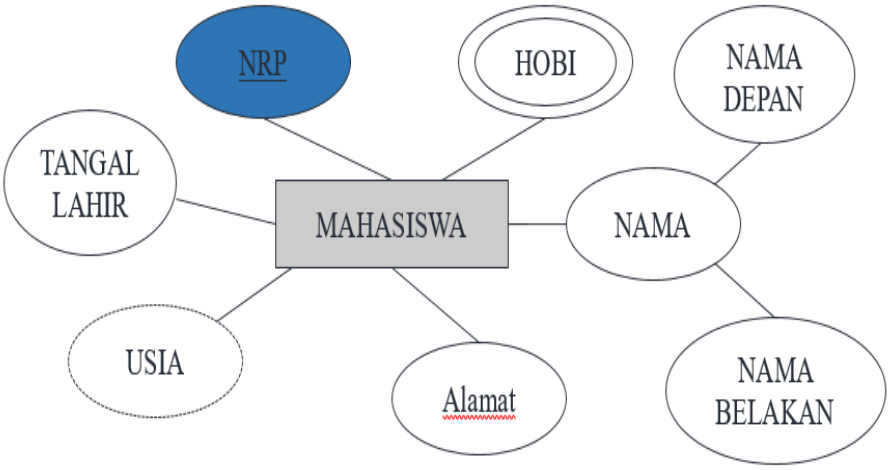
Aturan menggambar atribut :

1) Atribut dinyatakan dengan simbol ellips
2) Atribut di hubungkan dengan sebuah baris

Dalam diagram ER-D terdapat beberapa jenis atribut, dimana masing-masing atribut memiliki fungsi dan karakteristik yang berbeda-beda. Berikut adalah jenis simbol dalah ER-D yaitu :

A. Atribut Key

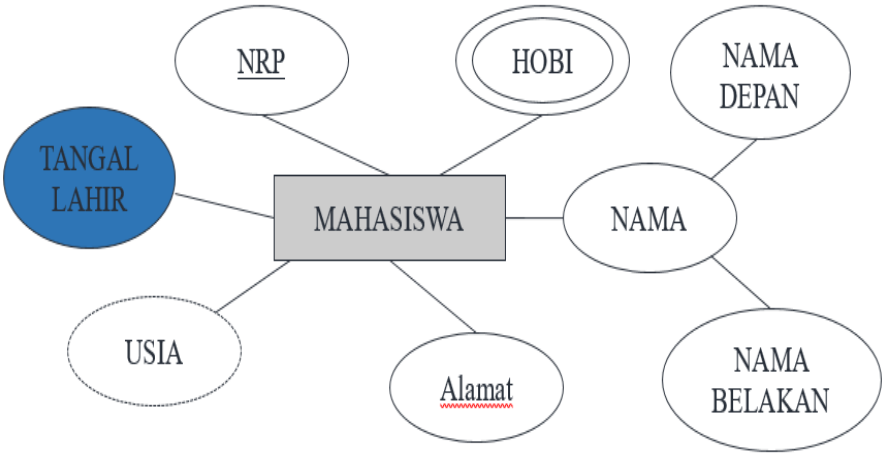
- Atribut yang dugunakan untuk menentukan suatu entitas secara unik.
- Nama atribut diberi garis bawah.



Gambar 5-14: Atribut Key

B. Atribut Simple

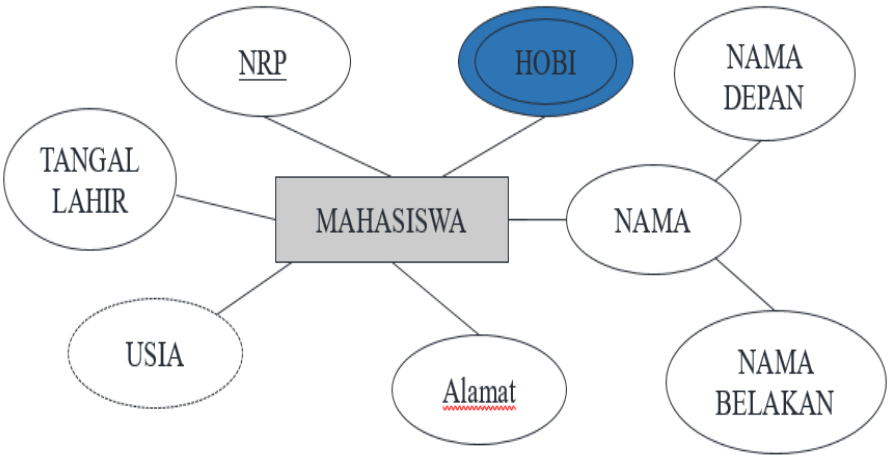
- Atribut bernilai tunggal
- Atribut terkecil yang tiak bisa dipilah lagi



Gambar 5-15: Atribut Simple

C. Atribut Multi Value

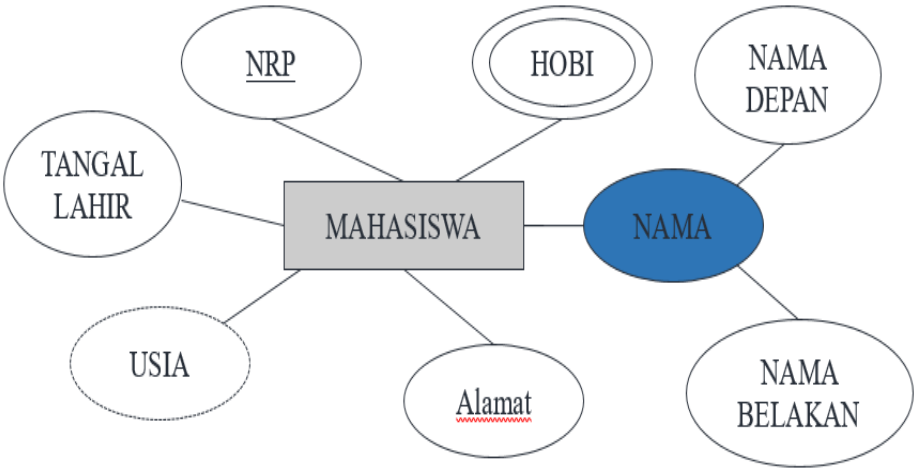
Atribut yang memiliki sekelompok nilai untuk setiap instan entitas.



Gambar 5-16: Atribut Multi Value

D. Atribut Composite

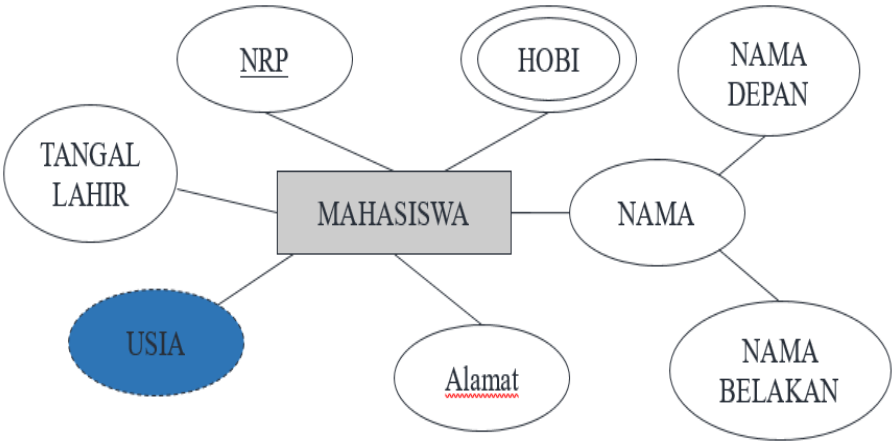
Atribut yang terdiri dari beberapa atribut yang lebih kecil yang mempunyai arti tertentu.



Gambar 5-17: Atribut Composite

E. Atribut Derivative

Atribut yang dihasilkan oleh atribut yang lain, dan diberi garis putus-putus.



Gambar 5-18: Atribut Derivative

5.3. Tahapan Membuat ER-D

Cara membuat atau mendesign *Entity Relationship Diagram*:

1. Identifikasi Entitas	Identifikasi peran, kejadian/kegiatan, lokasi, hal abstrak/konsep yang datanya disimpan oleh *end-user*
2. Tentukan *Relationship*	Tentukan hubungan antara sepasang entitas menggunakan relationship matriks.
3. Gambar "KASAR" ERD	Gambarkan entitas-entitas dan relationship diantara entitas untuk menghubungkannya.
4. Tentukan kardinalitas	Tentukan kardinalitas (pemunculan sutau entitas di entitas lainnya yang berhubungan)
5. Tentukan *Primary Key*	Identifikasi atribut data yang secara "unik" mengidentifikasi setiap entitas
6. Gambar ERD berdasar atribut kunci	Sertakan *primary key* di setiap entitas
7. Identifikasi atribut lainnya	Kumpulkan informasi detil yang penting dalam sistem yang sedang di kembangkan.
8. Petakan atribut	Untuk setiap atribut, letakkan dalam satu entitas yang tepat. Cari juga atribut yang ada dalam *relationship*.
9. Gambarkan ERD lengkap dengan atribut	Sesuaikan ERD hasil langkah 6 dengan entitas atau *relationship* hasil langkah 8
10. Periksa hasil	Apakah *Entity Relationship Diagram* akhir telah secara tepat mencerminkan data sistem ?

5.4. Logical Record Structured (LRS)

LRS adalah representasi dari struktur record-record pada tabel-tabel yang terbentuk dari hasil relasi antar himpunan entitas.

Menentukan Kardinalitas, Jumlah Tabel dan Foreign Key (FK).

1. One To One

   Contoh :

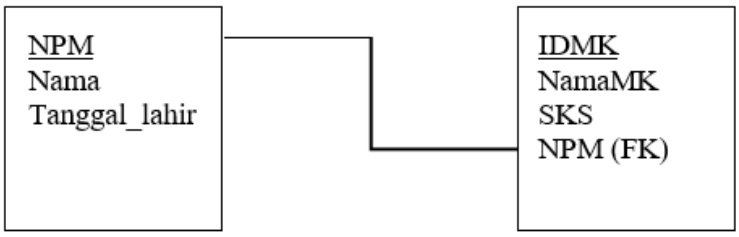
Gambar 5-19: Contoh One To One

Gambar diatas menunjukan satu orang pegawai dapat terlibat dalam satu atau lebih dari satu proyek, dan satu proyek dapat dikerjakan oleh satu atau lebih dari satu pegawai. Relasi one to one diatas akan membentuk 2 tabel :

a. Tabel Mahasiswa (NPM, Nama, Tanggal_lahir)

b. Tabel Matakuliah (IDMK, NamaMK, SKS)

LRS yang terbentuk adalah :



Gambar 5-20: LRS One To One

2. One To Many

   Contoh :

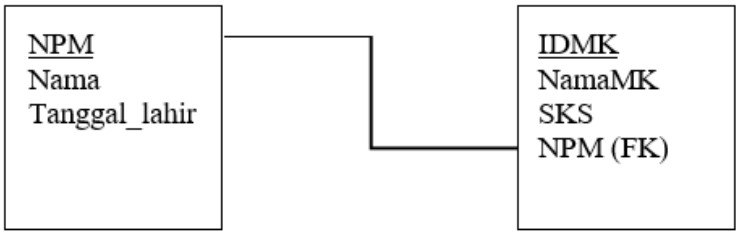
Gambar 5-21: LRS One To Many

Gambar diatas menunjukan satu orang mahasiswa dapat mengambil satu atau lebih dari satu matakuliah, sedangkan satu matakuliah hanya bisa diikuti oleh satu orang mahasiswa. Relasi one to Many diatas akan membentuk 2 tabel (tidak jauh beda dengan one to one):

a. Tabel Mahasiswa (NPM, Nama, Tanggal_lahir)

b. Tabel Matakuliah (IDMK, NamaMK, SKS)

LRS yang terbentuk adalah :



Gambar 5-22: LRS One To Many

3. Many To Many

Contoh :

Gambar 5-23: Contoh Many To Many

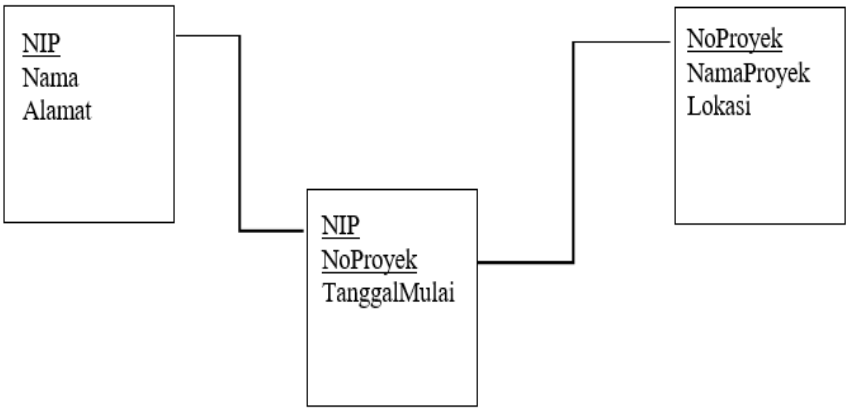
Gambar diatas menunjukan dimana satu orang pegawai dapat terlibat dalam satu atau lebih dari satu proyek, dan satu proyek dapat dikerjakan oleh satu atau lebih dari satu pegawai. Relasi Many to Many akan membentuk tiga buah tabel :

a. Pegawai (NIP, Nama, Alamat)

b. Proyek (NoProyek, NamaProyek, Lokasi)

c. Pengerjaan (NIP, NoProyek, TanggalMulai)

LRS yang terbentuk adalah



Gambar 5-24: LRS Many To Many

BAB VI
TEKNIK NORMALISASI

6.1. Pengertian Normalisasi

Ketika kita akan melakukan perancangan basis data maka tujuan utama dalam pengembangan model dan logikanya adalah untuk menciptakan gambaran data yang tepat bagi hubungan maupun batasannya. Merancang basis data yang tepat diperlukan agar kita bisa memiliki basis data yang kompak dan efisien dalam ruang penyimpanan, cepat dalam pengaksesan dan mudah dalam melakukan manipulasi data. Teknik yang digunakan untuk membantu merancang basis data tersebut yaitu dengan normalisasi.

Teknik normalisasi pertama kali diperkenalkan oleh tokoh basis data E.F. Codd pada tahun 1972. Normalisasi sering digunakan untuk melakukan serangkaian tes pada sebuah relasi, apakah relasi tersebut sudah sesuai atau melanggar persyaratan bentuk normal tertentu.

Pengertian normalisasi menurut E.F. Codd, adalah :

- Normalisasi digunakan untuk membuat struktur tabel (relasi) dalam basis data untuk mengurangi kemubaziran (tidak efisien) data.
- Terkadang dipakai sebagai perangkat verivikasi terhadap tabel-tabel yang dihasilkan dalam metodologi lain.
- Memberikan panduan yang sangat membantu bagi pengembang untuk mencegah penciptaan struktur tabel yang tidak efisien.

Sedangkan menurut Kroenke menyampaikan beberapa pengertian normalisasi sebagai berikut :

- Normalisasi sebagai proses untuk mengubah suatu relasi yang memiliki masalah tertentu ke dalam 2 buah relasi atau lebih yang tidak memiliki masalah tersebut (anomali).
- Secara garis besar, dapat disimpulkan normalisasi adalah sebuah proses yang digunakan untuk membentuk struktur basis data agar terhindar dari ambiguitas (Redudancy) sehingga lebih efisien.

Dari pengertian normalisasi dari beberapa tokoh basis data diatas, dapat disimpulkan bahwa pengertian normalisasi adalah proses yang berkaitan dengan model data relational untuk mengorganisasi himpunan data dengan ketergantungan dan keterkaitan yang tinggi atau erat. Hasil dari proses normalisasi adalah himpunan-himpunan data (table-table) dalam bentuk normal (normal form).

6.2. Tujuan Normalisasi

1. Memudahkan user dalam akses data
2. Optimalisasi struktur tabel
3. Optimalisasi storage
4. Mengurangi redundansi
5. Menghindari anomali (insert, delete, update)
6. Peningkatan integritas data

6.3. Anomaly

Anomaly merupakan penyimpangan-penyimpangan atau error atau inkonsistensi data yang terjadi pada saat dilakukan proses insert, delete maupun update dalam suatu basis data. Terdapat 3 jenis Anomaly sebagai berikut :

1. Insertion Anomaly

   Error yang terjadi sebagai akibat operasi insert record/tuple pada sebuah relation.

2. Deletion Anomaly

   Error yang terjadi sebagai akibat operasi delete record/tuple pada sebuah relation.

3. Update Anomaly

   Error yang terjadi sebagai akibat inkonsistensi data sebagai akibat dari operasi update record/tuple dari sebuah relation.

Selain banyak keuntungan dari normalisasi, ternyata normalisasi memiliki beberapa masalah pada relation yang sudah dinormalisasi, yaitu :

1. Performance Problem

   Masalah terhadap performa database

2. Referential Integrity Problem

   Masalah yang terjadi terhadap referensi antar data diantara dua tabel atau lebih.

6.4. Atribut dan Ketergantungan Fungsi

Ada beberapa konsep yang harus diketahui dalam melakukan normalisasi, yaitu :

1. Atribut Kunci

   Atribut kunci yang terdapat dalam database adalah :

   a. *Super key*
   b. *Candidate key*
   c. *Primary key*
   d. *Alternate key*
   e. *Foreign key*

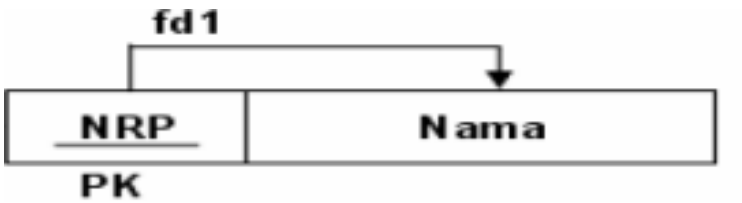
2. Ketergantungan Fungsi

   Untuk dapat melakukan normalisasi, harus bisa menentukan terlebih dahulu fungsional dependent (ketergantungan fungsional), khususnya dalam melakukan dekomposisi database. Beberapa jenis ketergantungan fungsi dalam normalisasi :

   a. Ketergantungan Fungsional (Functional Dependent)

      Keterkaitan antara hubungan dua atribut dalam sebuah relasi. Disimbolkan A→B : artinya B memiliki ketergantungan dengan A. berarti A secara fungsional menentukan B, atau B secara fungsional tergantung pada A.

Contoh :



NRP→Nama : Nama tergantung dengan NRP, atau NRP dapat menentukan nilai dari Nama.

b. Ketergantungan Funsional Penuh (Fully Functional Dependent)

Atribut Y pada relasi R dikatakan tergantung fungsional penuh pada stribut X pada relasi R, jika X tidak tergantung pada subset dari X (bila X adalah key gabungan).

Contoh :

No-pem	Na-pem	No-bar	Jumlah
P01	Baharu	B01	1000
P01	Baharu	B02	1500
P01	Baharu	B03	2000
P02	Sinar	B03	1000
P03	Harapan	B02	2000

Gambar 6-1: Contoh Full Functional Dependent

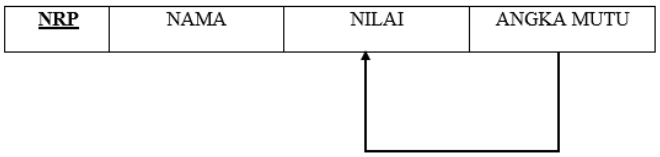
No-pem, No-Bar → Jumlah : jumlah tergantung penuh terhadap key nya. Artinya jika dalam sebuah relasi memiliki lebih dari satu key, dan

atribut bukan key tergantung terhadap semua key yang ada, maka atribut tersebut memiliki ketergantungan fungsional penuh.

c. Ketergantungan Transitif

Atribut Z pada relasi R dikatakan tergantung transitif pada atribut X, atau menjadi atribut biasa pada suatu relasi tetapi menjadi kunci pada relasi lain.

Contoh :



Nilai → AngkaMutu : artinya atribut angka mutu tergantung terhadap atribut nilai, atribut nilai merupakan atribut biasa atau bukan kunci.

d. Ketergantungan Parsial

Ketergantungan pasial adalah suatu atribut hanya tergantung pada salah satu atau sebagian kunci atau primary key saja.

Contoh :

No-pem	Kode-kota	Kota	No-bar	Jumlah
P01	1	Jakarta	B01	1000
P01	1	Jakarta	B02	1500
P01	1	Jakarta	B03	2000
P02	3	Bandung	B03	1000
P03	2	Surabaya	B02	2000

No-Pem → Kode-Kota: merupakan ketergantungan parsial, karena kode-kota hanya tergantung ada salah satu kunci yang ada.

6.5. Tahapan Normalisasi

Aturan-aturan normalisasi dinyatakan dengan istilah bentuk normal. Bentuk normal adalah suatu aturan yang dikenakan pada relasi-relasi dalam basis data dan harus dipenuhi oleh relasi-relasi tersebut pada level-level normalisasi. Beberapa bentuk dari tahapan normalisasi adalah :

Bentuk Normal Pertama (1NF)

↓

Bentuk Normal Kedua (2NF)

↓

Bentuk Normal Ketiga (3NF)

↓

Bentuk Normal Boyce-Codd (BCNF)

↓

Bentuk Normal Keempat (4NF)

↓

Bentuk Normal Kelima (5NF)

Untuk urutan langkah-langkah dari setiap tahapan normalisasi (bentuk normalisasi) adalah sebagai berikut :

1. Bentuk Normal Pertama (1NF)

   Suatu relasi memenuhi 1-NF jika dan hanya jika setiap atribut dari relasi tersebut hanya memiliki nilai tunggal dalam satu baris atau record.

Lakukan :

- ❑ Buat ***Un-Normalized Form*** (bentuk tidak normal) dengan mengidentifikasi seluruh item data.
- ❑ Tentukan ***primary key*** dari relasi tersebut.
- ❑ Tentukan ***grup berulang*** (item data atau atribut yang memiliki nilai lebih dari satu dalam suatu baris data/record).
- ❑ Tentukan ***key*** dari ***grup berulang***.
- ❑ ***Coret derived atribut*** (atribut yang nilainya dapat diabaikan, karena dihasilkan dari atribut lain)
- ❑ Pisahkan ***grup berulang*** menjadi suatu relasi baru dengan membawa key atribut dari relasi awal.

2. Bentuk Normal Kedua (2NF)

Suatu relasi memenuhi *2-NF* jika dan hanya jika :

- ❑ Memenuhi 1-NF.
- ❑ Setiap atribut yang bukan kunci utama tergantung secara *fungsional* terhadap semua atribut kunci dan bukan hanya sebagian atribut.

Lakukan :

- Perhatikan relasi yang memiliki key atribut lebih dari 1.
- Pisahkan atribut-atribut yang tergantung hanya pada *sebagian / salah satu key* bersama dengan key pengidentifikasi atribut tersebut menjadi suatu relasi baru.

3. Bentuk Normal Ketiga (3NF)

Suatu relasi memenuhi bentuk III (3-NF) jika dan hanya jika :

- ❑ Relasi tersebut memenuhi 2-NF.
- ❑ Setiap atribut bukan kunci tidak tergantung secara fungsional kepada atribut bukan kunci yang lain dalam relasi tersebut.

<u>Lakukan :</u>

- Pisahkan atribut-atribut yang tergantung pada atribut yang bukan key bersama dengan atribut bukan key tersebut menjadi suatu relasi baru.

4. Bentuk Normal Boyce-Codd (BCNF)

- ❑ Secara praktis, tujuan rancangan database adalah cukup sampai level 3NF. Akan tetapi untuk kasus-kasus tertentu kita bisa mendapatkan rancangan yang lebih baik lagi apabila bisa mencapai ke BCNF.
- ❑ BCNF ditemukan oleh : R.F. Boyce dan E.F. Codd
- ❑ Suatu relasi R dikatakan dalam bentuk BCNF : Jika dan hanya jika setiap atribut kunci (Key) pada suatu relasi adalan kunci Kandidat (Candidate Key). Kunci Kandidat (Candidate Key) adalah atribut-atribut dari entitas yang mungkin dapat digunakan sebagai Kunci (Key) atribut.
- ❑ BCNF hampir sama dengan 3NF, dengan kata lain setiap BCNF adalah 3NF.

5. Bentuk Normal Keempat (4NF)

Bentuk normal keempat berkaitan dengan sifat Ketergantungan Banyak-Nilai (Multivalued Depedency) pada suatu tabel yang merupakan pengembangan dari ketergantungan fungsional.

6. Bentuk Normal Kelima (5NF)

Bentuk normal kelima merupakan nama lain dari ProjectJoin Normal Form (PNJF) yaitu berhubungan dengan ketergantungan relasi antar tabel (Join Dependency).

BAB VII
BAHASA QUERY FORMAL

7.1. Pengantar Bahasa Query Formal

Bahasa query merupakan bahasa yang termasuk dalam kategori bahasa tingkat tinggi yang digunakan user untuk mendapatkan informasi atau data dari basis data. Bahasa query dikelompokkan menjadi dua, yaitu : Query procedural dan Query non Procedural.

Dalam bahasa query formal, ada dua dasar pembentukan bahasa query :

1. Aljabar Relasional

   Merupakan salah satu bahasa manipulasi untuk database relasional. Aljabar relasional merupakan kumpulan operasi terhadap relasi dimana setiap operasi menggunakan satu atau lebih relasi untuk menghasilkan satu relasi yang baru.

2. Kalkulus relasional

   Merupakan bahasa manipulasi teoritis yang non procedural. Artinya bahasa ini mengekspresikan “what to do or what characteristics are required”dan tidak mengekspresikan “how to do it” seperti pada bahasa query procedural. Kalkulus relasional dilandasi dengan teori predicate calculus yang menggunakan fungsi sebagai suatu ekspresi logic.

7.2. Operator Aljabar Relasional

Operator pada aljabar relational dibagi menjadi 2 kelompok:

1. Operator dasar untuk fundamental operational

Operator dasar terdiri dari : selection, projection, cartesian product, set difference dan union.

2. Operator tambahan untuk additional operasional

Operator tambahan terdiri dari setvintersection, theta join, natural join dan division.

Contoh :

Tabel dibawah ini adalah contoh untuk mengerjakan perintah-perintah Aljabar relasional:

Relasi Mahasiswa

NIM	NAMA_MHS	ALAMAT	J_KEL
1105090222	HAFIDZ	DEPOK	LAKI-LAKI
1105091002	RAFFA	DEPOK	LAKI-LAKI
1105095000	NAIA	DEPOK	PEREMPUAN
1104030885	ARIF	P.LABU	LAKI-LAKI
1206090501	LENI	KMP. MELAYU	PEREMPUAN
1206090582	WAHYUNI	TANGERANG	PEREMPUAN
1205097589	ARIS	DEPOK	LAKI-LAKI
1106094586	YANI	CILEDUG	PEREMPUAN
110709	BAMBANG	SALEMBA	LAKI-LAKI

A. Operator Dasar

1. Selection (σ) Lower Case Omega

Operasi selection menyeleksi tupel-tupel pada sebuah relation yang memenuhi predicate/syarat yang sudah ditentukan.

Sintaks : σ predicate/syarat $^{(tabel)}$

Contoh 1 :

- Tampilkan data mahasiswa yang beralamat di depok

  Query : σ alamat=”depok” $^{(mahasiswa)}$

Contoh 2 :

- Tampilkan data mahasiswa yang beralamat di depok atau Jenis Kelamin Perempuan.

  Query : σ alamat=”depok” V j_Kel=”perempuan” $^{(mahasiswa)}$

2. Projection (π)

Operator projection beroperasi pada sebuah relation, yaitu membentuk relation baru dengan mengcopy atribute-atribute dan domain-domain dari relation tersebut berdasarkan argumen-argumen pada operator tersebut.

Sintask : π A1,A2, …..An $^{(nama\ tabel)}$; dimana A adalah Atribut.

Contoh :

- Tampilkan Nama dan alamat dari mahasiswa

  Query : π Nama, alamat $^{(mahasiswa)}$

3. Cartesian product (X)

Operator ini merupaka binary operation yaitu operator yang beroperasi pada dua relasi. Operator Cartesian product menggunakan symbol X.

Sintaks : R1 X R2, dimana R adalah relasi.

Contoh :

- Tampilkan hasil dari kartesian tabel A dan tabel B

  Query : tabel A X tabel B

4. Union (∪)

Operasi untuk menghasilkan gabungan tabel dengan syarat kedua tabel memiliki atribut yang sama yaitu domain atribut ke-i masing-masing tabel harus sama RUS={ X I X E R atau X E S}.

Contoh :

Sintaks yang digunakan dalam operasi union ini adalah :

$$\mathbf{R \cup S = \{x \mid x \in R \text{ atau } X \in S\}}$$

5. Set difference (-)

Operasi untuk mendapatkan tabel disuatu relasi tapi tidak ada di relasi lainnya. Dengan kata lain operator ini berfungsi untuk mengeliminasi entity atau record dari suatu tabel yang ada pada tabel yang lainnya.

R – S = { X I X E R dan X E S }.

B. Operator Tambahan

1. Set Intersection (∩)

Operasi untuk menghasilkan irisan dua tabel dengan syarat kedua tabel memiliki atribut yang sama, domain atribut ke-i kedua tabel tersebut sama.

Sintaks : R = R1∩ R2; dan akan menghasilkan relasi R dengan elemen yang terdapat di R1 dan juga terdapat di R2.

2. Theta Join

Operasi yang menggabungkan operasi cartesian product dengan operasi selection dengan suatu kriteria.

Contoh : $S1_{S1.No_id < R1.No_id} R1$

3. Natural Join

Operasi menggabungkan operasi selection dan cartesian product dengan suatu kriteria pada kolom yang sama.

Contoh : $S1\ \theta_{No_id}\ R1$

4. Division

Merupakan operasi pembagian atas tuple-tuple dari 2 relation.

BAB VIII
BAHASA QUERY TERAPAN

8.1. Structured Query Language (SQL)

SQL adalah singkatan dari Structured Query Language. Sedangkan pengertian SQL adalah suatu bahasa (language) yang digunakan untuk mengakses data di dalam sebuah database relasional. SQL sering juga disebut dengan istilah query.

Pada tahun 1999 dikeluarkan standar baru yaitu SQL99 atau disebut juga SQL99. Saat ini sebenarnya tidak ada server basis data yang 100% mendukung SQL99. Hal ini disebabkan masing-masing server memiliki dialek masing-masing. Subdivisi SQL:

a. Data Definition Language (DDL)

Query-query ini digunakan untuk mendefinisikan struktur atau skema basis data.

b. Data Manipulation Language (DML)

Query-query ini digunakan untuk manajemen data dalam basis data.

8.2. Pengelompokan Perintah Structured Query Language (SQL)

a. Data Definition Language (DDL)

Digunakan untuk mendefinisikan (membuat, menghapus, mengubah) objek dalam database. Yang termasuk kedalam kelompok perintah DDL adalah CREATE, DROP dan ALTER.

1. CREATE

Create berfungsi untuk membuat objek (database, tabel, dll) dalam sistem database.

a) Membuat database

Perintahnya adalah : CREATE DATABASE nama_database

Contoh : CREATE DATABASE Penjualan

b) Membuat tabel

Perintahnya adalah : CREATE TABLE nama_tabel (kolom tipedata constraint, kolom2 tipedata2 constaint2,…..).

Contoh : CREATE TABLE Mahasiswa (NPM INT Primary Key, Nama Varchar(20)).

Keterangan :

(1) Nama tabel merupakan nama tabel yang baru, panjangnya tidak dapat lebih dari 8 karakter, tidak memakai spasi, berisi huruf maupun angka.

(2) Nama Kolom adalah nama untuk kolom yang baru, panjangnya tidak dapat lebih dari 10 karakter, tidak memiliki spasi, berisi huruf dan angka.

(3) Tipe Data adalah jenis data yang nilainya akan dimasukkan dalam kolom yang telah ditentukan.

2. DROP

Drop berfungsi menghapus Objek dalam database.

Perintahnya : DROP<spasi>Jenis_Objek<spasi>nama_objek

- Contoh menghapus database : DROP DATABASE Penjualan
- Contoh menghapus table : DROP TABLE Mahasiswa

3. ALTER

Alter berfungsi merubah Objek dalam database. Alter yang akan dibahas pada bagian ini adalah Alter yang diterapkan pada tabel.

a) Menambah kolom pada tabel

ALTER TABLE Nama_tabel

ADD nama_kolom tipedata

Contoh :

ALTER TABLE Mahasiswa

ADD Alamat Varchar(20)

b) Menghapus kolom pada tabel

ALTER TABLE Nama_tabel

DROP COLUMN nama_kolom

Contoh :

ALTER TABLE Mahasiswa

DROP COLUMN Alamat

c) Merubah tipe data atau panjang data suatu kolom

ALTER TABLE Nama_tabel

ALTER COLUMN nama_kolom tipedata

Contoh :

ALTER TABLE Mahasiswa

ALTER COLUMN Alamat Varchar(50)

b. Data Manipulation Language (DML)

Digunakan untuk melakukan manipulasi (perubahan) data yang ada dalam tabel. Yang termasuk kedalam kelompok perintah DML adalah INSERT, SELECT, UPDATE dan DELETE.

1. INSERT

   Insert berfungsi untuk menambah nilai (data) pada suatu tabel.

   Perintahnya : INSERT INTO tabel VALUES (nilai1, nilai2, ….).

   Contoh :

   Masukan satu data kedalam tabel mahasiswa :

   INSERT INTO MAHASISWA VALUES (123456,”AGUS”)

2. SELECT

   Select berfungsi untuk mengambil nilai (data) pada suatu tabel. Ada tiga bentuk select :

   a) Mengambil seluruh data

      SELECT * FROM namatabel

      Contoh : SELECT * FROM Mahasiswa

      Keterangan : tanda (*) mewakili semua baris dan semua kolom.

   b) Mengambil kolom tertentu

      SELECT kolom1, kolom2,… FROM namatabel

      Contoh : SELECT NPM, NAMA FROM Mahasiswa

   c) Mengambil baris data tertentu

      SELECT * FROM namatabel WHERE Kriteria

      Contoh : SELECT * FROM Mahasiswa WHERE Nama=”kokom”

Keterangan : klausa WHERE berfungsi untuk melakukan filter baris data pada tabel sesuai dengan kriteria yang ditetapkan.

3. UPDATE

Update berfungsi untuk merubah nilai (data) pada suatu tabel. Perintah update adalah :

UPDATE nama tabel SET nama field = nilai baru

Contoh : UPDATE Mahasiswa SET Nama=”Komar”

Contoh 2 : UPDATE Mahasiswa SET Nama=”Komar” Where NPM=1234

4. DELETE

Delete berfungsi untuk menghapus nilai (data) pada suatu tabel. Perintanya adalah :

DELETE * FROM nama_Tabel

Contoh1 : DELETE FROM Mahasiswa

Contoh 2 : DELETE DROM Mahasiswa WHRE Nama=”Agus”

DAFTAR PUSTAKA

Connolly, T., & Begg, C. (2015). *Database Systems : A Practical Approach to Design, Implementation, and Management, 6th Edition.* Boston: Pearson Education.

Fathansyah. (2015). *Basis Data - Revisi Kedua.* Bandung: Informatika.

Indrajani. (2015). *Database Design (Case Study All In One).* Jakarta: PT. Elex Media Komputindo.