Bab 1

Review Database 1

POKOK BAHASAN:

- ✓ Pendahuluan
- ✓ ER-Model
- ✓ Model Relasional
- ✓ Structured Query Language
- ✓ Normalisasi

TUJUAN BELAJAR:

Setelah mempelajari materi dalam bab ini, mahasiswa diharapkan mampu:

- ✓ Memahami *Database Management System* dan komponen utamanya
- ✓ Memahami ER-Model dan dapat menggunakannya sebagai desain awal dari database
- ✓ Memahami SQL dan apa saja yang tercakup dalam bahasa SQL
- ✓ Memahami konsep normalisasi dan dapat melakukan normalisasi data

1.1. PENDAHULUAN

Pada saat sekarang ini, kesuksesan suatu organisasi bergantung pada kemampuannya menangkap data secara akurat dan tepat waktu, dalam hal pengoperasian, pengaturan data secara efektif, maupun penggunaan data untuk keperluan analisis.

Kemampuan untuk mengatur atau mengolah sejumlah data, dan kecepatan untuk mencari informasi yang relevan, adalah aset yang sangat penting bagi suatu organisasi. Untuk mendapatkan himpunan data yang besar dan kompleks, user harus memiliki alat

bantu (*tools*) yang akan menyederhanakan tugas manajemen data dan mengekstrak informasi yang berguna secara tepat waktu.

Basis data adalah kumpulan data, yang dapat digambarkan sebagai aktifitas dari satu atau lebih organisasi yang berelasi. Sebagai contoh, basis data universitas berisi informasi mengenai :

Entiti , semisal mahasiswa, fakultas, mata kuliah, dan ruang kelas

Relasi diantara entitas, seperti pengambilan kuliah yang dilakukan oleh mahasiswa, staf pengajar di fakultas, dan penggunaan ruang perkuliahan.

Manajemen Sistem Basis Data (*Database Management System – DBMS)* adalah perangkat lunak yang didesain untuk membantu dalam hal pemeliharaan dan utilitas kumpulan data dalam jumlah besar. DBMS dapat menjadi alternatif penggunaan secara khusus untuk aplikasi, semisal penyimpanan data dalam file dan menulis kode aplikasi yang spesifik untuk pengaturannya.

Tujuan dari pengajaran mata kuliah basis data adalah untuk memberikan suatu pendahuluan mengenai sistem manajemen basis data, dengan penekanan pada baagimana cara mengorganisasi suatu informasi dalam DBMS, untuk memelihara informasi tersebut dan melakukan pengambilan informasi secara efektif, dan bagaimana cara mendesain suatu basis data dan menggunakan suatu DBMS secara efektif pula. Penggunaan DBMS untuk suatu aplikasi tergantung pada kemampuan dan dukungan DBMS yang beroperasi secara efisien. Sehingga agar bisa menggunakan DBMS dengan baik, perlu diketahui cara kerja dari DBMS tersebut. Pendekatan yang dilakukan untuk menggunakan DMBS secara baik, meliputi implementasi DBMS dan arsitektur secara mendetail untuk dapat memahami desain dari suatu basis data.

1.2. ER-MODEL

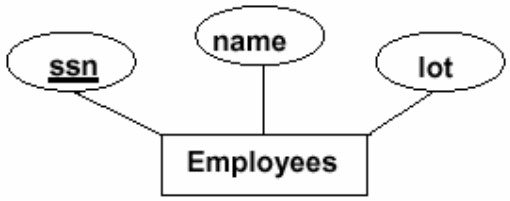
Pada ER Model, gambaran dunia nyata diistilahkan dalam obyek dan relasinya. ER model biasa digunakan untuk mengembangkan inisial dari desain basis data. ER model menyediakan suatu konsep yang bermanfaat yang dapat mengubah deskripsi informal dari apa yang diinginkan oleh user menjadi hal yang lebih detail, presisi, dan deskripsi detail tersebut dapat diimplementasikan ke dalam DBMS.

Pada konteks yang lebih luas, ER model digunakan dalam fase desain basis data konseptual.

1.2.1. ENTITI, ATRIBUT, DAN HIMPUNAN ENTITI

Entiti adalah obyek dunia nyata yang dapat dibedakan dari obyek yang lain. Entiti digambarkan (dalam basis data) dengan menggunakan himpunan atribut. Himpunan entiti yang sejenis disimpan dalam himpunan entiti.

Himpunan entity : Kumpulan entity yang sejenis.



Gambar 1-1: *Entiti Pegawai (Employee)*

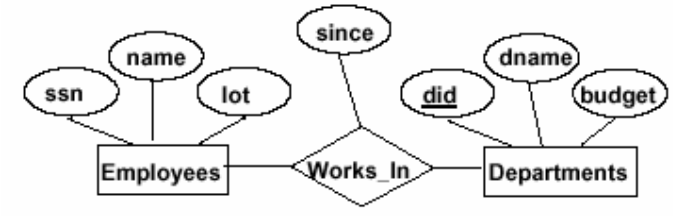
Misal : himpunan data pegawai

- Semua entity dalam himpunan entity memiliki himpunan atribut yang sama
- Tiap himpunan entity memiliki kunci (key)
- Tiap atribut memiliki domain.

1.2.2. RELASI DAN HIMPUNAN RELASI

Relasi adalah asosiasi diantara dua atau lebih entity

Misal : Ani bekerja di Departemen Farmasi

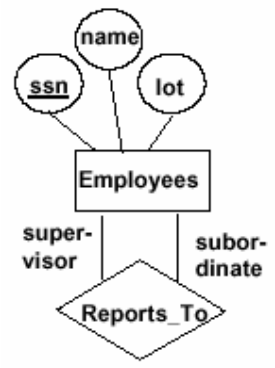


Gambar 1-2: *Relasi antar Entiti*

Himpunan Relasi : Himpunan dari relasi-relasi yang sejenis

Himpunan relasi n-ary *R* berelasi dengan sejumlah himpunan entity n E1 … En

Himpunan entity yang sama dapat berpartisipasi dalam himpunan relasi yang berbeda, atau mempunyai peran yang berbeda dalam suatu himpunan yang sama.



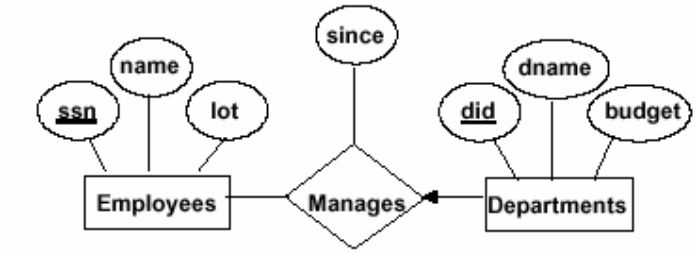
Gambar 1-3: *Self Relationship*

1.2.3. FITUR TAMBAHAN UNTUK ER-MODEL

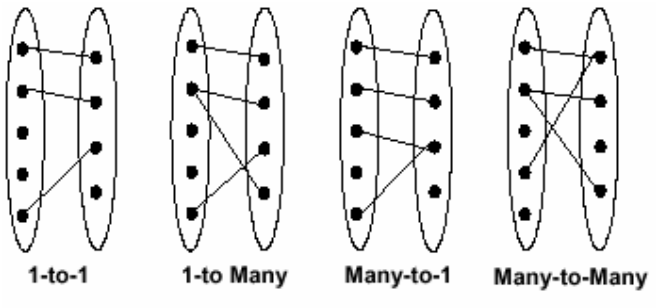
Berikut ini dibahas beberapa fitur tambahan untuk ER-Model :

Batasan Kunci (Key Constraints)

- Pada suatu contoh kasus, seorang pegawai dapat bekerja pada beberapa departments; sebuah departement memiliki banyak pegawai
- Sebaliknya, tiap departement hanya memiliki seorang manager, yang berhubungan dengan key constraint pada *Manages.*



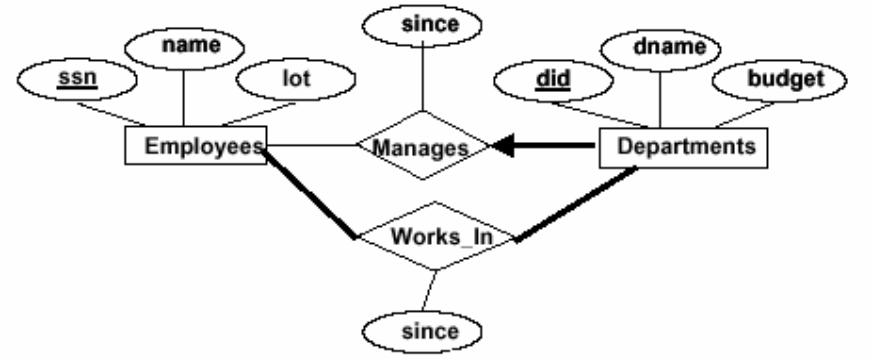
Gambar 1-4: *Contoh Key Constraint antar Entiti*



Gambar 1-5: *Macam-macam Key Constraint*

Batasan Partisipasi (Participation Constraints)

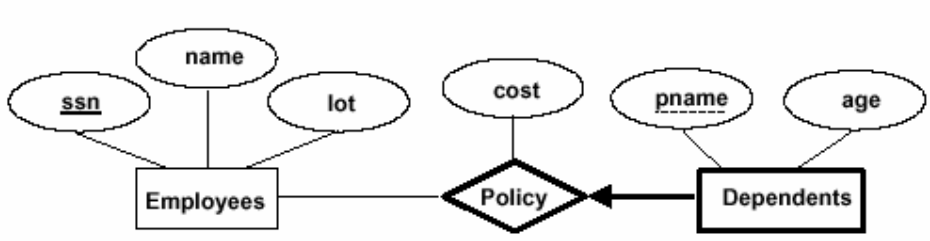
- Apakah setiap departemen mempunyai seorang manager ?
    - Jika semua departemen pasti mempunyai manager maka partisipasi *Departements* dalam *Manages* dapat dikatakan total. Sebaliknya jika tidak semua departement memiliki manager maka partisipasinya adalah partial.



Gambar 1-6: ***Contoh Participation Constraint***

Entiti Lemah (Weak Entity)

- Entiti lemah dapat diidentifikasi secara unik jika terdapat peran kunci utama (primary key) yang berasal dari atau dimiliki oleh entity yang lain (owner).
    - Himpunan entity owner dan entity lemah harus berartisipasi dalam himpunan relasi one-to-many (satu owner, banyak entity lemah).

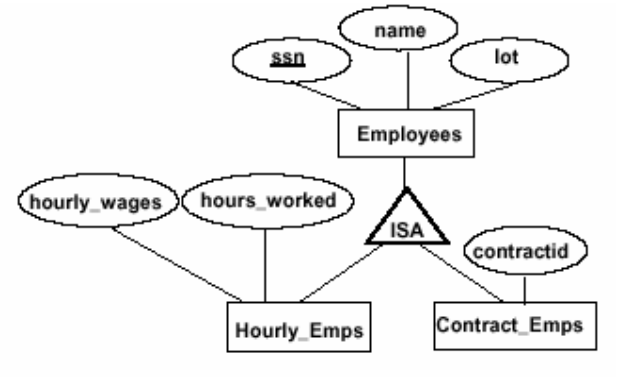


Gambar 1-7: ***Contoh Weak Entity***

1.2.4. HIRARKI KLAS

Seperti pada C++, dan bahasa pemrograman yang lain, suatu atribut dapat diturunkan. Jika kita deklarasikan A ISA B, setiap entity A juga termasuk entity B.

- *Overlap constraints* : Bolehkah seorang pegawai mempunyai status sebagai pegawai dengan hitungan gaji perjam (Hourly_Emps) sama halnya seperti pegawai dengan perjanjian kontrak (Contract_Emps) ? (Boleh/Tidak)
- *Covering constraints :* Apakah setiap entity *Employees* juga merupakan entity *Hourly_Emps* dan *Contract_Emps* ?



Gambar 1-8 : ***Hirarki Klas***

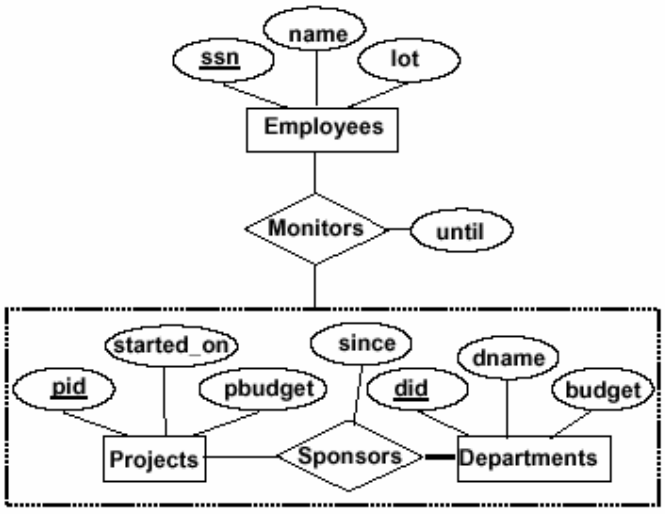
Alasan menggunakan ISA :

- Untuk menambahkan deskripsi atribut yang lebih spesifik pada subclass.
- Untuk mengidentifikasi entity yang berpartisipasi dalam suatu relasi.

1.2.5. AGGREGASI

Aggregasi digunakan pada saat kita perlu memodelkan apa saja yang terlibat dalam suatu himpunan relasi. Aggregasi membolehkan kita untuk memperlakukan suatu himpunan relasi sebagai himpunan entity untuk tujuan partisipasi dalam relasi yang lain.

Gambar berikut menunjukkan bahwa *Monitors* adalah relasi yang distinct dengan deskripsi atribut. Juga dapat dikatakan bahwa tiap *sponsorship* dimonitor oleh seorang pegawai.



Gambar 1-9: ***Contoh Aggregasi***

1.3. MODEL RELASIONAL

Basis Data Relasional adalah himpunan relasi. Suatu relasi adalah himpunan kolom atau tupel (semua barisnya bersifat distinct/unik).

Sedangkan relasi itu sendiri terdiri dari dua bagian yaitu :

- *Instance :* table dengan baris dan kolom
  #baris = kardinalitas, #kolom/fields = degree/arity
- *Skema :* menentukan nama relasi, plus nama dan tipe kolom

Contoh relasi misal :

Students(sid : string, name : string, login : string, age : integer, gpa : real).

sid	name	login	age	gpa
53666	Jones	jones@cs	18	3.4
53688	Smith	smith@eecs	18	3.2
53650	Smith	smith@math	19	3.8

Gambar 1-10 : *Contoh Instance dari Relasi Students*

Pada gambar, contoh instance dari relasi *Students* memiliki kardinalitas = 3, degree = 5, semua baris bersifat distinct. (*Pertanyaan :* Apakah semua kolom dalam instance relasi juga harus distinct ?)

Kekuatan utama dari model relasional adalah kesederhanaannya, dan kelebihannya adalah dalam melakukan query atas data. Query dapat ditulis secara intuitif, dan DBMS bertanggungjawab untuk mengevaluasinya secara efisien.

Kita dapat melakukan query pada beberapa table yang saling berelasi. Contoh pada table berikut jika terdapat table *Enrolled* yang berelasi dengan table *Students* sebelumnya dengan key field *sid* :

sid	cid	grade
53831	Carnatic101	C
53831	Reggae203	B
53650	Topology112	A
53666	History105	B

Kemudian diberikan query :

```
SELECT S.name, E.cid
FROM Students S, Enrolled E
WHERE S.sid=E.sid and E.grade=”A”
```

Maka table yang dihasilkan dari query tersebut adalah :

S.name	E.cid
Smith	Topology112

Yaitu mencari data *Students* (nama *Students* dan mata kuliah yang diikutinya) yang mendapat nilai “A”.

1.3.1. BATASAN INTEGRITAS (INTEGRITY CONSTRAINT)

Batasan Integritas adalah suatu kondisi yang harus bernilai benar untuk suatu instance dalam basis data, misal : batasan domain

- Dispesifikasi saat skema didefinisikan
- Diperiksa pada saat suatu relasi dimodifikasi

Instance dari relasi disebut legal jika bisa memenuhi semua batasan integritas (integrity constraints) yang telah dispesifikasi. Batasan integritas juga digunakan untuk menghindari kesalahan dari entry data

Berikut akan dibahas satu persatu batasan integritas dalam model relasional.

Batasan Kunci Primer (Primary Key Constraints)

Himpunan suatu fields merupakan suatu key dari suatu relasi jika :

- Tidak ada dua tupel yang distinct yang mempunyai nilai yang sama untuk semua key fields, dan
- Key tersebut tidak memiliki subset.
    - Pernyataan 2 salah ? bagaimana dengan *superkey*
    - Jika terdapat lebih dari satu key untuk suatu relasi, maka salah satu dari key tersebut akan dipilih oleh DBA untuk menjadi primary key.

Misal : *sid* adalah key untuk relasi *Students. (*Bagaimana dengan *name),* himpunan key (*sid,gpa*) adalah merupakan superkey.

Primary dan Candidate Key dalam SQL :

- Dari kemungkinan banyak candidate keys (dispesifikasi dengan menggunakan UNIQUE), salah satunya dapat dipilih menjadi *primary key.*
- Seorang *Students* dapat mengambil suatu course dan hanya menerima satu nilai untuk *grade* dari course yang diikutinya.

Berikut contoh penggunaan batasan kunci primer :

```
CREATE TABLE Enrolled
( sid CHAR(20),
cid CHAR(20),
grade CHAR(2),
PRIMARY KEY (sid,cid)
```

```
CREATE TABLE Enrolled
(sid CHAR(20),
cid CHAR(20),
grade CHAR(2),
PRIMARY KEY (sid)
UNIQUE(cid,grade))
```

Foreign Keys

Foreign key adalah himpunan fields dalam satu relasi yang digunakan untuk melakukan referensi ke tupel pada relasi yang lain (Harus berkorespondensi dengan *primary key* pada relasi yang kedua). Berlaku seperti *logical pointer*

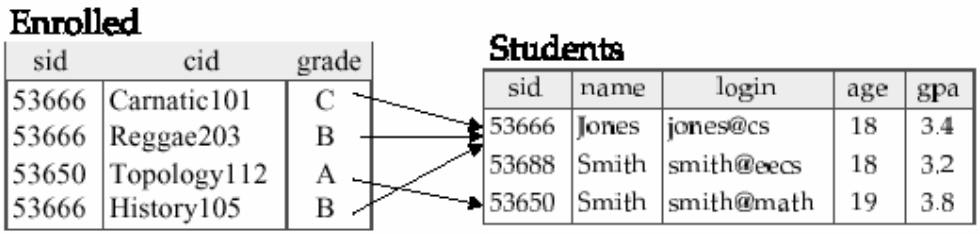
Misal *sid* adalah *foreign key* yang direfer dari relasi *Students* :

- Enrolled(sid : string, cid : string, grade : string)

Foreign Keys dalam SQL :

- Hanya *Students* yang terdaftar dalam relasi *Students* yang diperbolehkan untuk mengikuti suatu perkuliahan (course).

```
CREATE TABLE Enrolled
(sid CHAR(20), cid CHAR(20), grade CHAR(2),
 PRIMARY KEY(sid,cid),
 FOREIGN KEY(sid) REFERENCES Students)
```



Enrolled

sid	cid	grade
53666	Carnatic101	C
53666	Reggae203	B
53650	Topology112	A
53666	History105	B

Students

sid	name	login	age	gpa
53666	Jones	jones@cs	18	3.4
53688	Smith	smith@eecs	18	3.2
53650	Smith	smith@math	19	3.8

Referential Integrity

Misal pada relasi *Students* dan *Enrolled; sid* dalam *Enrolled* adalah foreign key yang mereferensi relasi *Students.*

Apa yang harus dilakukan jika tupel *Enrolled* dengan suatu data *Students* yang tidak terdaftar dalam relasi *Students* disisipkan ? (Hindari hal ini).

Apa yang harus dilakukan jika tupel *Students* di-hapus ?

- Hapus juga semua tupel *Enrolled* yang merefer ke tupel *Students* yang dihapus tersebut
- Tidak mengijinkan dilakukan penghapusan jika tupel tersebut merefer ke tupel pada relasi yang lain (alternatif lain dari yang pertama)
- Ubah *sid* dalam tupel *Enrolled* menjadi *default sid (*alternatif yang lain lagi).
- (Dalam SQL, juga dapat dilakukan setting pada tupel *Enrolled* yang direfer oleh tupel *Students* yang dihapus tersebut dengan memberikan nilai khusus yaitu *null,* yang artinya 'tidak diketahui' (*unknown* atau *inapplicable).*

Sama halnya jika primary key dari tupel Students dilakukan perubahan (update).

SQL/92 mendukung pilihan berikut untuk perintah delete dan update :

- Default-nya adalah tidak dilakukan apa-apa (pembatalan perintah delete/update).
- CASCADE (juga men-delete semua tupel yang merefer ke tupel yang di-delete).
- Set nilai NULL/DEFAULT (Set nilai foreign key dari tupel yang direferensi).

Contoh pembuatan referential integrity :

```
CREATE TABLE Enrolled
(sid : CHAR(20),
cid : CHAR(20),
grade : CHAR(2),
PRIMARY KEY(sid,cid),
FOREIGN KEY(sid)
REFERENCES Students
ON DELETE CASCADE
ON UPDATE SET DEFAULT)
```

1.4. *STRUCTURED QUERY LANGUAGE*

Structured Query Language (SQL) adalah bahasa database relasional yang dibuat berdasarkan suatu standart. Bentuk dasar dari SQL adalah sebagai berikut :

SELECT [DISTINCT] select-list
FROM from-list
WHERE qualification

Setiap query dalam SQL harus memiliki klausa SELECT, yang menentukan kolom yang akan ditampilkan pada hasil, dan klausa FROM yang menentukan cross product table. Klausa optional WHERE menentukan syarat-syarat seleksi pada table yang ditunjukkan oleh FROM.

Berikut ini akan dibahas sintaksis query SQL dasar dengan lebih mendetail :

- **from list** pada klausa FROM adalah daftar nama table. Nama tabel dapat diikuti oleh nama alias; nama alias berguna ketika nama tabel yang sama muncul lebih dari sekali pada from list
- **select-list** adalah daftar nama kolom (termasuk ekspresinya) dari tabel-tabel yang tercantum pada form list. Nama kolom dapat diawali dengan nama alias dari tabel.
- **Kualifikasi** pada klausa WHERE merupakan kombinasi boolean atau pernyataan kata sambung logika dari kondisi yang menggunakan eksepresi yang melibatkan operator pembanding. Sedangkan ekspresi itu sendiri dapat berupa nama kolom, konstanta atau aritmatika dan string.
- Kata kunci **distinct** bersifat pilihan yang menghapus duplikat dari hasil query.

SQL menyediakan tiga konstruksi set-manipulation yang memperluas query dasar, yaitu UNION, INTERSECT dan EXCEPT. Juga operasi set yang lain seperti : IN (untuk memeriksa apakah elemen telah berada pada set yang ditentukan), ANY dan ALL (untuk membandingkan suatu nilaid engan elemen pada set tertentu), EXISTS (untuk memeriksa apakah suatu set kosong atau isi). Operator IN dan EXISTS dapat diawali dengan NOT.

Fitur SQL yang lain yaitu NESTED QUERY, artinya query yang memiliki query lain di dalamnya, yang disebut dengan subquery. Nested query digunakan jika terdapat suatu nilai yang tidak diketahui (*unknown values).*

SQL mendukung lima operasi aggregat yang diterapkan pada sembarang kolom yaitu :

- COUNT : untuk menghitung cacah
- SUM : menghitung jumlah seluruh nilai
- AVG : menghitung rata-rata nilai
- MAX : mencari nilai paling besar
- MIN : mencari nilai paling kecil.

Kadangkala operasi aggregat diperlukan pada sekeompok grup dari baris pada relasi. Untuk menulis query semacam itu, dibutuhkan klausa GROUP BY. Dan penambahan klausa HAVING jika kita ingin menerapkan suatu kondisi terhadap data yang sudah dikelompokkan dengan GROUP BY.

1.5. NORMALISASI

Normalisasi adalah perbaikan skema database. Latar belakang diperlukannya normalisasi adalah karena adanya penyimpanan informasi yang redundan.

Istilah normalisasi berasal dari E.F. codd, salah seorang perintis teknologi basis data. Normalisasi adalah proses untuk mengubah suatu relasi tertentu ke dalam dua buah relasi atau lebih.

Berikut ini akan dijelaskan proses Normalisasi sampai dengan bentuk normal ketiga.

Bentuk Normal Pertama (1NF)

Suatu relasi dikatakan dalam bentuk normal pertama jika dan hanya jika setiap atribut bernilai tunggal untuk setiap atribut bernilai tunggal untuk setiap baris contoh:

Tabel 1. sebelum bentuk normal pertama

NIP	Nama	Hoby
10113024	Endang C Permana	Olahraga
		Baca Buku
10113025	Samsul	Dengar Musik
		Makan

Table 2. yang sudah dalam bentuk normal pertama

NIP (Primary Key)	Nama	Hoby
10113024	Endang C Permana	Olahraga
10113024	Endang C Permana	Baca Buku
10113025	Samsul	Dengar Musik
10113025	Samsul	Makan

Bentuk Normal Kedua (2NF)

Suatu relasi dikatakan dalam bentuk normal kedua jika berada dalam normal pertama dan setiap atribut bukan kunci memiliki ketergantungan sepenuhnya terhadap kunci primer

contoh:

Tabel 3. sebelum bentuk normal kedua

NIP (Primary Key)	Nama	Kd_Mata_kuliah	Nilai
10113024	Endang C Permana	001	70
10113024	Endang C Permana	002	90
10113025	Samsul	003	100
10113025	Samsul	004	60

Table 4. yang sudah dalam bentuk normal kedua

NIP (Primary Key)	Kd_Mata_kuliah (Primary Key)	Nilai
10113024	001	70
10113024	002	90
10113025	003	100
10113025	004	60

Table 5.

NIP (Primary Key)	Nama
10113024	Endang C Permana
10113025	Samsul

Bentuk Normal Ketiga (3NF)

Suatu relasi dikatakan dalam bentuk normal ketiga jika berada dalam normal kedua dan setiap atribut bukan kunci tidak memiliki ketergantungan transitif terhadap kunci primer contoh:

Tabel 6. sebelum bentuk normal ketiga

Kode_proyek	Nama	Alamat_kota
001	Endang C Permana	Bandung
002	Endang C Permana	Ebandung
003	Samsul	Jakarta
004	Samsul	Jakarta

Table 7. yang sudah dalam bentuk normal ketiga

Kode_Proyek	Nama
001	Endang C Permana
002	Endang C Permana
003	Samsul
004	Samsul

Table 8.

Nama	Alamat_kota
Endang C Permana	Bandung
Samsul	jakarta

RINGKASAN:

- Basis data adalah kumpulan data, yang dapat digambarkan sebagai aktifitas dari satu atau lebih organisasi yang berelasi.
- Manajemen Sistem Basis Data (***Database Management System – DBMS)*** adalah perangkat lunak yang didesain untuk membantu dalam hal pemeliharaan dan utilitas kumpulan data dalam jumlah besar.
- Pada ER Model, gambaran dunia nyata diistilahkan dalam obyek dan relasinya dan digunakan untuk mengembangkan inisial dari desain basis data.

- Kelebihan dari model relasional adalah kesederhanaannya dalam melakukan query atas data. Query dapat ditulis secara intuitif, dan DBMS bertanggungjawab untuk mengevaluasinya secara efisien.
- Batasan Integritas adalah suatu kondisi yang harus bernilai benar untuk suatu instance dalam basis data
- *Structured Query Language (SQL)* adalah bahasa database relasional yang dibuat berdasarkan suatu standart, dan memiliki bentuk dasar :

  SELECT [DISTINCT] select-list
  FROM from-list
  WHERE qualification
- Normalisasi adalah perbaikan skema database yang dibuat dengan tujuan untuk menghindari penyimpanan informasi yang redundan.

LATIHAN SOAL :

1. Gambarlah sebuah diagram ER yang mengungkapkan informasi ini.

Perusahaan rekaman Notown memutuskan untuk menyimpan semua informasi mengenai musisi yang mengerjakan albumnya (seperti halnya data perusahaan lain) dalam sebuah database. Pihak perusahaan menyewa anda sebagai desainer database (dengan biaya konsultasi sebesar $2.500 / hari).

- Tiap musisi yang melakukan rekaman di Notown mempunyai SSN, nama, alamat dan nomer telpon. Para musisi yang dibayar lebih rendah akan mendapatkan alamat yang sama dengan musisi lain, dan satu alamat mempunyai satu nomer telpon.
- Tiap instrumen yang digunakan untuk merekam berbagai macam lagu di Notown mempunyai nama (contoh : gitar, sinthesizer, flute) dan kunci musik (contoh : C, B-flat, E-flat).
- Tiap album yang dicatata di Notown mempunyai judul rekaman, tanggal copyright, format (contoh : CD atau MC) DAN SEBUAH INDENTIFIKASI ALBUM.

- Tiap lagu yang di catat di Notown mempunyai judul dan pengarang lagu
- Tiap musisi mungkin memainkan beberapa instrumen, dan tiap instrumen dapat dimainkan oleh beberapa musisi
- Tiap album mempunyai beberapa lagu di dalamnya tapi tidak ada lagu yang muncul bersamaan dalam satu album.
- Tiap lagu dibawakan oleh satu atau lebih musisi dan seorang musisi bisa membawakan beberapa lagu.
- Tiap album dibawakan seorang musisi yang berperan sebagai produser. Seorang musisi bisa menghasilkan beberapa album.

2. Perhatikan skema relasional berikut ini :

Emp(*eid:***integer,** ***ename :*** **string**, *age :* **integer,** *salary:* **real**)

Works(*eid:***integer,** *did:***integer,** *pct_time:* **integer**)

Dept(*did:***integer,** *dname:* **string,** *budget:* **real,** *managerid:* **integer**)

Berikan contoh constraint foreign key yang melibatkan relasi Dept. Apa saja pilihan yang ada untuk melaksanakan constraint ini pada saat user berusaha untuk menghapus record pada Dept ?

3. Untuk skema relasional pada nomer 2, definisikan relasi Dept pada SQL sehingga setiap department dipastikan memiliki seorang manajer.
4. Untuk skema relasional pada nomer 2, tuliskan pernyataan SQL untuk menampilkan karyawan yang bekerja di department ‘IT’.
5. Untuk skema relasional pada nomer 2, tuliskan pernyataan SQL untuk menampilkan karyawan yang bekerja di department ‘IT’ dan memiliki usia yang lebih dari usia rata-rata orang-orang yang bekerja di department ‘IT’
6. Lakukan normalisasi data pada tabel Kuliah yang memiliki atribut : kode kuliah, nama kuliah, sks, semester, nama dosen, waktu kuliah, ruang.