DATABASE
Entity Relasionsip Diagram

Satrio Agung W, Ari Kusyanti dan Mahendra Data
Teknik Informatika, Fakultas Teknik,
Universitas Brawijaya, Email : informatika@ub.ac.id

MODUL

3

ENTITY RELATIONSHIP DIAGRAM (ERD)

Pendahuluan

Entity Relasionalship Diagram adalah suatu model penyajan data dengan menggunakan Entity dan Relasionship. Tujuan dari penyajan ini adalah agar database dapat dipahami dan dicancang dengan mudah.

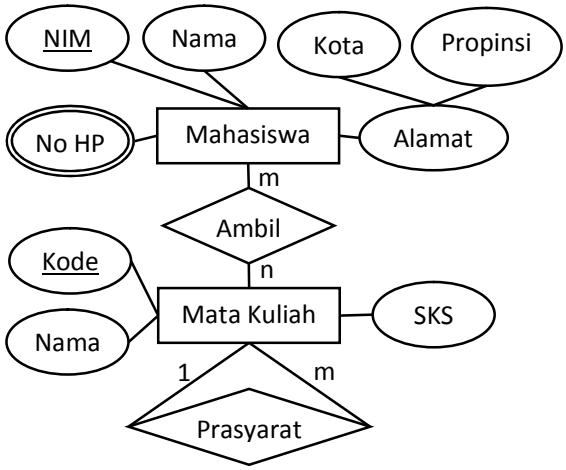
Tujuan

Setelah membaca modul ini, mahasiswa diharapkan dapat memahami :

- Konsep ERD
- Cara penyajian ERD
- Membuat ERD
- Menganalisa ERD

Entity Relationship Diagram

Entity Relationship Diagram adalah suatu model penyajan data dengan menggunakan Entity dan Relasionship. ERD menggambarkan model konseptual untuk menggambarkan struktur logis dari basisdata berbasis grafis. Tujuan dari penyajan ini adalah agar database dapat dipahami dan dicancang dengan mudah.



Gambar 3.1 Model data hirarki

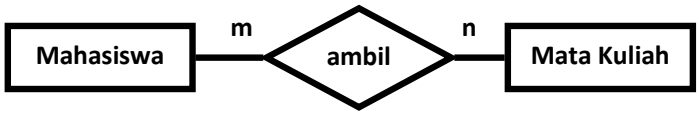
Symbol ERD

Entity

Entity adalah objek yang dapat dibedakan dalam dunia nyata. Sedangkan Entity Set adalah kumpulan dari entitty yang sejenis. Entity set dapat berupa objek secara fisik (Rumah, Kendaraan, Peralatan) atau objek secara konsep(Pekerjaan, Perusahaan). Entity disimbolkan dengan persegi panjang (▭).

Relationship

Relationship adalah hubgan yang terjadi antara satu atau lebih entity. Sedangkan Relationship set adalah kumpulan relationship yang sejenis. Relationship disimbolkan dengan jajar genjang (◇).



Gambar 3.2 Contoh relasi antara Mahasiswa dengan Mata Kuliah

Attribute

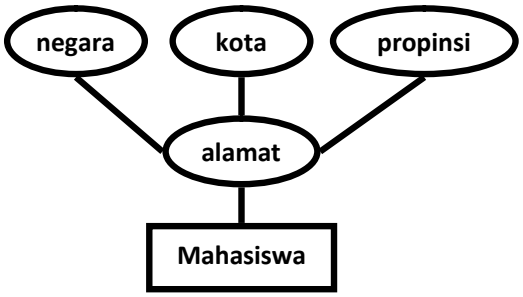
Attribute adalah karakteristik dari tiap entity atau relationship yang menyediakan penjelasan detail mengenai entity atau relationship tersebut. Nilai dari attribute adalah data aktual atau informasi yang disimpan pada suatu attribut di dalam entity atau relationship, dimana tiap attribute memiliki domain (value set) tersendiri. Domain (value set) adalah batas-batas nilai yang diperbolehkan bagi suatu attribute. Attribute disimbolkan dengan jajar genjang (⬭).

Jenis-jenis attribute yang digunakan dalam ERD adalah:

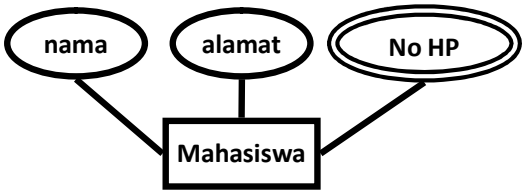
- *Key* : Atribut yang digunakan untuk menentukan suatu entity secara unik.
- *Atribut Simple* : Atribut sederhana yang tidak dapat dibagi dalam beberapa bagian.
- *Atribut Komposit* : Atribut yang dapat dibagi lagi dalam beberapa bagian. Contoh : Alamat yang dapat dibagi lagi menjadi Negara, Propinsi dan Kota
- *Atribut Single-valued* : Atribut yang memiliki paling banyak satu nilai untuk setiap baris data.

- *Multi-valued attributes* : Atribut yang dapat diisi dengan lebih satu nilai tetapi jenisnya sama. Contoh : Nomor Telp, Alamat, Gelar
- *Atribut Turunan* : Atribut yang diperoleh dari pengolahan dari atribut lain yang berhubungan. Contoh : Umur, IP
- *Attribute Key* : Attribut yang dapat dijadikan kunci untuk mecari data dalam relasi. Contoh: NIM Mahasiswa

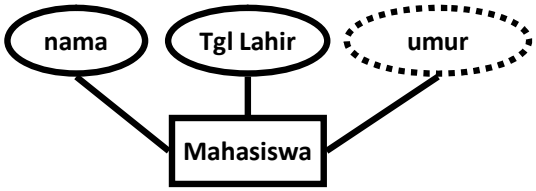
Contoh macam-macam attribute:



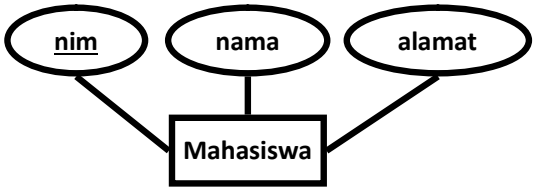
Gambar 3.3 Attribute Composite



Gambar 3.4 Multi Value Attribute



Gambar 3.4 Multi Value Attribute



Gambar 3.5 Key Attribute

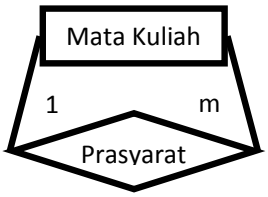
Derajat Relasi

Derajat Relasi menunjukan banyaknya himpunan entitas yang saling berelasi. Jenis derajat himpunan relasi adalah:

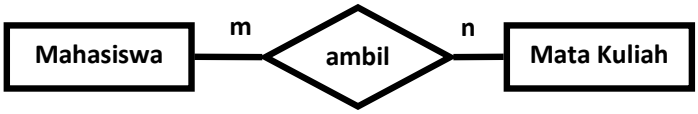
- *Unary Degree* (Derajat Satu) melibatkan sebuah entitas yang berelasi dengan dirinya sendiri
- *Binary Degree*(Derajat Dua) Himpunan relasi melibatkan dua himpunan entitas. Secara umum himpunan relasi dalam sistem basis data adalah binary

- *Ternary Degree* (Derajat Tiga) Himpunan relasi memungkinkan untuk melibatkan lebih dari dua himpunan entitas

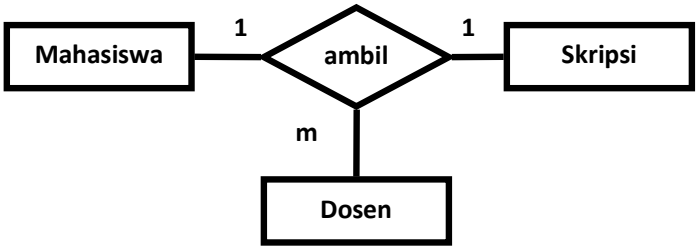
Contoh Derajat Relasi



Gambar 3.6 Unary Degree Relationship



Gambar 3.7 Binary Degree Relationship



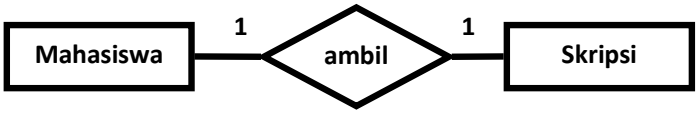
Gambar 3.8 Tenary Degree Relationship

Pemetaan Kardinalitas Relasi

Pemetaan kardinalitas relasi menggambarkan banyaknya jumlah maksimum entitas dapat berelasi dengan entitas pada himpunan entitas yang lain. Untuk Himpunan relasi biner pemetaan kardinalitasnya dapat merupakan salah satu dari tipe2 berikut :

1. Satu ke Satu (One to one)
   Artinya, sebuah entity hanya dapat berelasi dengan satu buah object di entity yang lain. Kardinalitas ini disimbolkan dengan 1-1
2. Satu ke Banyak (One to many), disimbolkan dengan 1-m
   Artinya, sebuah entity dapat berelasi dengan banyak object di entity yang lain. Kardinalitas ini disimbolkan dengan 1-m
3. Banyak ke Satu (Many to one), disimbolkan dengan m-1
   Ini adalah kebalikan dari one to many, maksudnya banyak entity akan berelasi dengan satu objek yang sama pada entity yang lain. Kardinalitas ini disimbolkan dengan 1-m
4. Banyak ke Banyak (Many to many), disimbolkan dengan m-n
   Artinya, akan ada banyak entity yang akan berelasi dengan banyak object di entity yang lain. Kardinalitas ini disimbolkan dengan m-n

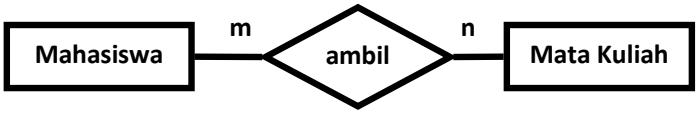
Contoh Penggambaran Pemetaan Kardinalitas Relasi



Gambar 3.9One to One

Gambar 3.10 One to Many

Gambar 3.11 Many to One



Gambar 3.12 Many to Many

Key

Penggunaan *key* merupakan cara untuk membedakan suatu entitas didalam himpunan entitas dengan entitas lain. Secara konsep, Masing-masing entitas (nilainya) berbeda, perbedaannya terlihat pada isi dari masing-masing atributnya. Oleh karena itu, dibutuhkan suatu atribut yang memiliki nilai yang menjadi pembeda dengan entitas lain. Key adalah satu atau gabungan dari beberapa atribut yang dapat membedakan semua row dalam relasi secara unik.

Ada 3 macam key yang dapat diterapkan pada suatu relasi :

1. *Super Key*
   Merupakan satu atau lebih atribut (kumpulan atribut) yang dapat membedakan satiap baris data dalam sebuah relasi secara unik.
2. *Candidate Key*
   Merupakan kumpulan atribut minimal yang dapat membedakan setiap baris data dalam sebuah relasi secara unik
3. *Primary Key*
   Merupakan salah satu dari candidate key yang terpilih
   Pemilihan primary key dari sejumlah candidate key umumnya didasari oleh :
   a. Key tersebut lebih sering (lebih natural) untuk dijadikan sebagai acuan
   b. Key tersebut lebih ringkas
   c. Jaminan keunikan key tersebut lebih baik

Contoh:

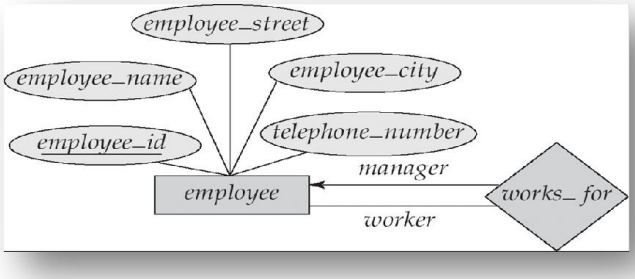
Mahasiswa = (NIM, NAMA_MHS, ALAMAT_MHS, TGL_LAHIR)

1. *Super Key*
   (NIM, NAMA_MHS, ALAMAT_MHS, TGL_LAHIR)
   (NIM, NAMA_MHS, ALAMAT_MHS)
   (NIM, NAMA_MHS)
   (NAMA_MHS)
   (NIM)

2. *Candidate Key*
   (NAMA_MHS, ALAMAT_MHS)
   (NIM)
3. *Primary Key*
   (NIM)

Peran (Role)

Relasi himpunan entitas tidak harus dalam bentuk yang berbeda, contoh label "manager" dan "worker" disebut Roles (peran), yang menspesifikasi bagaimana entitas employee berinteraksi melalui relasi *Works-for*. Peran dalam ER diagram diindikasikan dengan memberikan label (nama) pada garis yang menghubungkan relasi dengan entitas. Label peran bersifat optional dan digunakan untuk mengklarifikasi semantik suatu relasi.



Gambar 3.13 Peran

Dalam menggambarkan kardinalitas pada Diagram ER, digunakan :

- garis panah (→) yang menunjukkan "Satu" atau
- garis biasa (—) yang menunjukkan "Banyak", antara relasi dengan entitas

Weak Entity

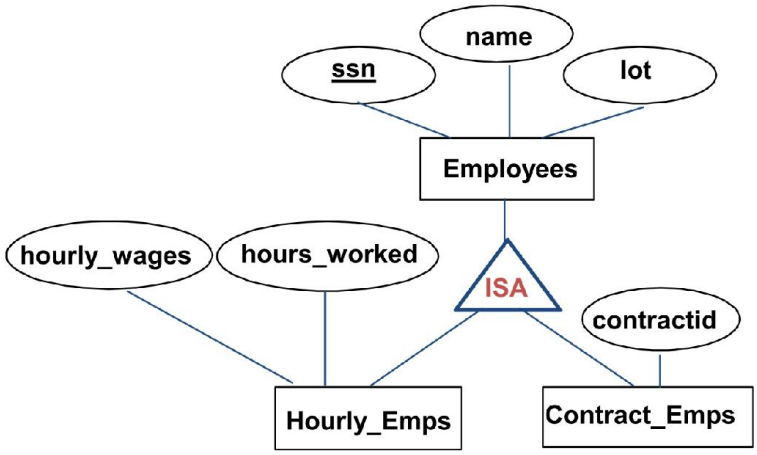
Weak Entity adalah suatu entity dimana keberadaan dari entity tersebut tergantung dari keberadaan entity lain. Entity yang merupakan induknya disebut Identifying Owner dan relationship-nya Disebut Identifyimg Relationship. Weak Entity Selalu mempunyai Total Participation Constraint dengan Identifying Owner. Contoh : entity tanggungan keberadaannya bergantung pada karyawan

Gambar 3.14 Weak Entity

ISA ('is a') Hirarkies

Seperti dalam C++, atau PLs lainnya, attribut diwariskan/diturunkan. Jika kita menyatakan A ISA B, tiap entitas A juga dipandang sebagai entitas B. Ada beberapa alasan menggunakan ISA, yaitu :

- Untuk menambahkan atribut deskriptif spesifik pada a subclass.
- Untuk mengidentifikasi entitas yang berpartisipasi dalam relationship



Gambar 3.15 Contoh ISA Hirarkies

Dalam ISA hirarkies terdapat beberapa *constrains*, yaitu:

- *Overlap constraints*: Satu objek entias tidak bisa terdapat dalam lebih dari satu entitas turunan dengan induk entitas yang sama. Contoh : Joe tidak dapat menjadi Hourly_Emps seperti juga entitas Contract_Emps? (lihat gambar 3.15)
- *Covering constraints*: Setiap objek dalam entitas induk harus masuk ke dalam salah satu entitas turunannya. Contoh : Tiap entitas Employees harus menjadi salah satu dari entitas Hourly_Emps atau Contract_Emps? (lihat gambar 3.15)

PUSTAKA

http://www.db-class.org/

http://infolab.stanford.edu/~ullman/dscb/gslides.html

http://infolab.stanford.edu/~ullman/fcdb.html

http://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-830-database-systems-fall-2010/