

MODUL 1

ENTITAS DAN ATRIBUT

Tujuan Instruksional Umum

Mahasiswa mampu mendeskripsikan, membedakan dan menentukan entitas dan atribut

Tujuan Instruksional Khusus

1. Mahasiswa dapat menentukan dan membedakan antara entitas dan atribut
2. Mahasiswa memahami dan menggunakan simbol dan terminology dalam ER diagram
3. Mahasiswa mampu mengidentifikasi dengan benar entitas yang dibutuhkan dari suatu bisnis proses tertentu
4. Mahasiswa mampu memahami dan menentukan jenis jenis atribut dan simbolnya
5. Mahasiswa mampu membedakan kebutuhan entitas, atribut atau nilai dari suatu atribut tertentu

MATERI PRAKTIKUM

1. ENTITAS

Entitas adalah benda, orang, tempat, unit, objek atau hal lainnya yang mempresentasikan suatu data.

Entities dapat berupa:

- Sesuatu yang berwujud, misalkan ORANG atau BARANG
- Sesuatu yang tidak berwujud, misalkan TINGKAT KEAHLIAN, UKURAN BAJU
- Event/kejadian, misalkan konser

Tujuan Entitas:

1. Dapat mengetahui bagaimana mengatur dan mengklasifikasikan data, memungkinkan untuk menarik kesimpulan yang berguna tentang suatu fakta.
2. Pada saat ini adanya sistem dan teknologi informasi, dapat menghasilkan sejumlah besar fakta yang membutuhkan struktur dan keteraturan.
3. Penting untuk mempelajari tentang entitas karena entitas adalah hal-hal yang akan disimpan datanya.

Sebagai contoh:

- Sekolah perlu menyimpan data tentang (minimal): SISWA, GURU, MATA PELAJARAN, RUANG, KELAS

2. ATRIBUT

Entitas mempunyai elemen yang disebut atribut, dan berfungsi mendeskripsikan karakter dari entitas Seperti entitas, atribut mewakili sesuatu yang penting bagi bisnis.

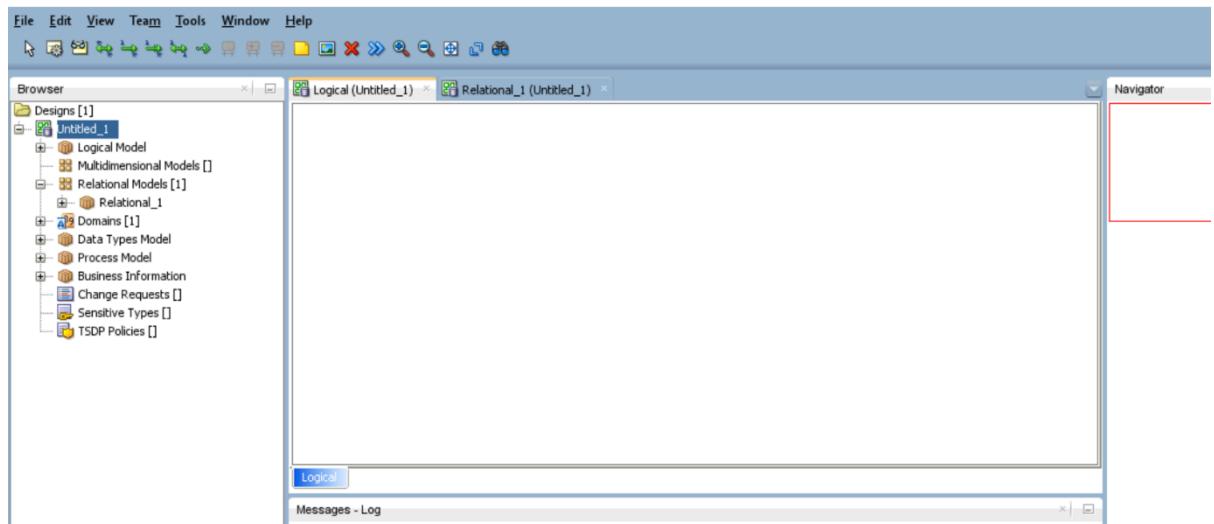
Atribut adalah informasi khusus yang membantu:

- menjelaskan entitas
- Hitung entitas
- Memenuhi syarat suatu entitas
- Klasifikasikan entitas
- Menentukan entitas

Setiap entitas dapat terdiri dari beberapa atribut contohnya entitas karyawan maka memiliki atribut nama, karyawan, alamat, dan id.

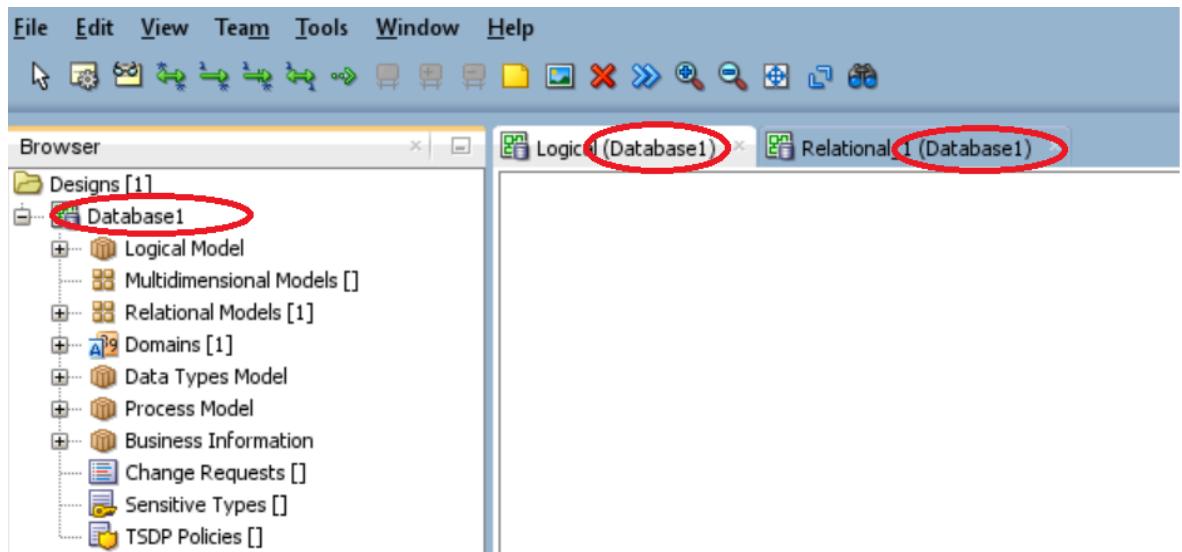
Cara Menggambarkan Entitas dan Atribut di Data Modeler

1. Buka Data Modeler

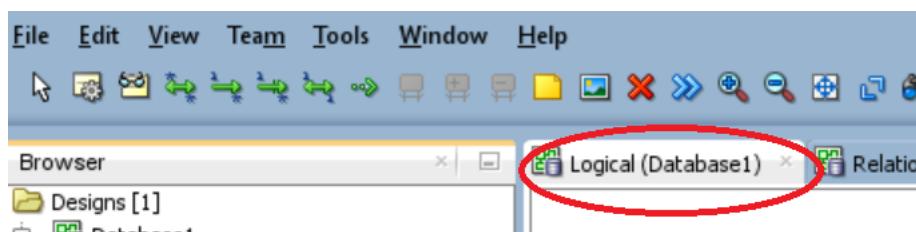


- Untuk memberi nama basis data yang anda buat, klik File, Pilih Save As, Ketikkan nama basis data yang anda buat, maka tulisan Untitled_1 akan berubah sesuai nama yang anda buat.

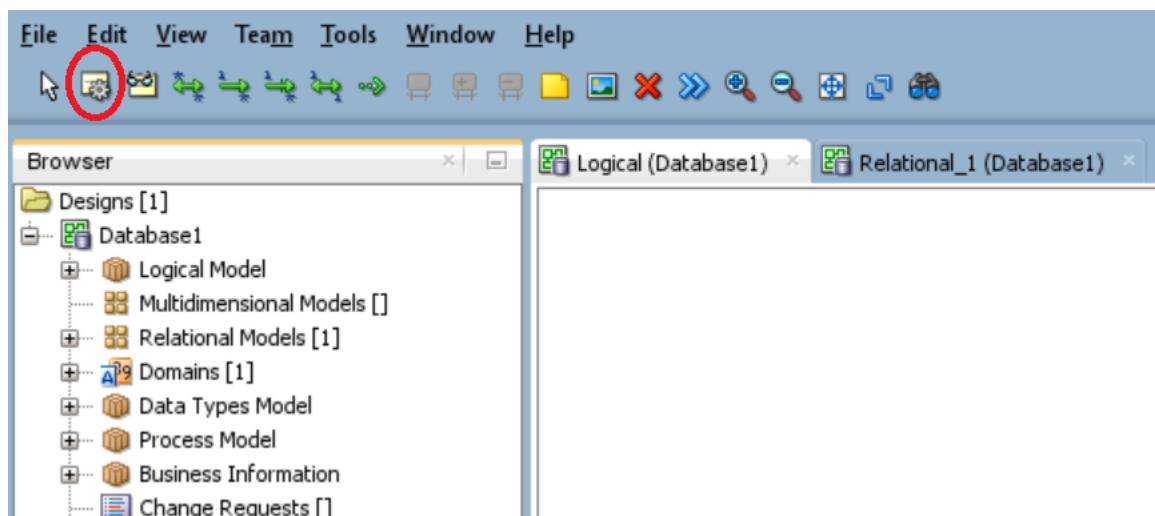
Misal Save As Database1, maka akan menjadi sebagai berikut:



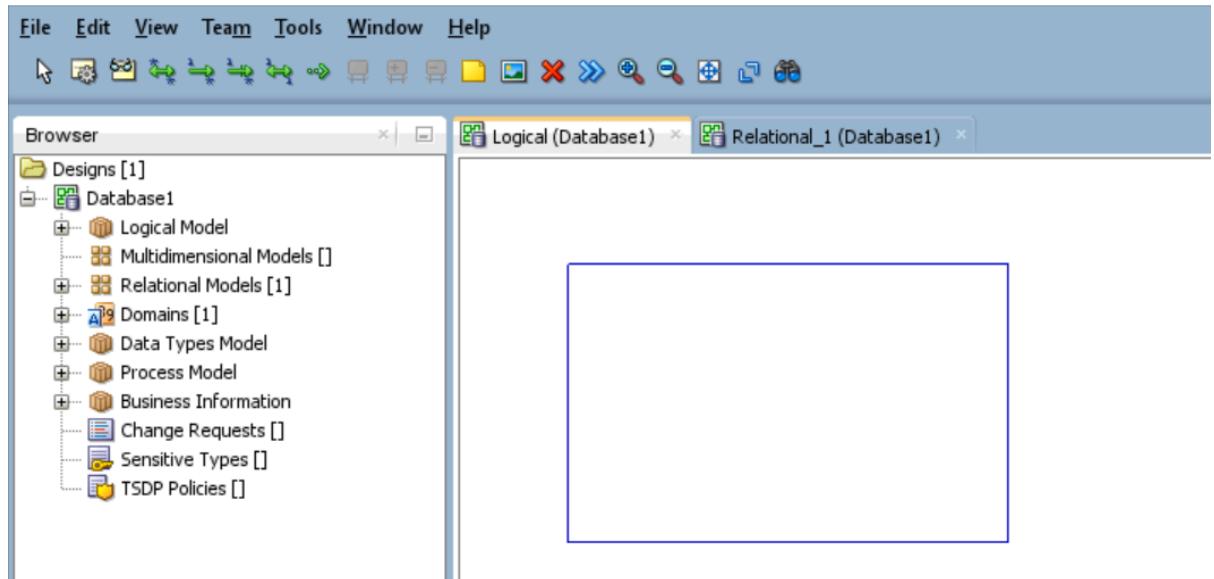
- Untuk memulai mendesain model konseptual, klik di layer Logical (Database_1)



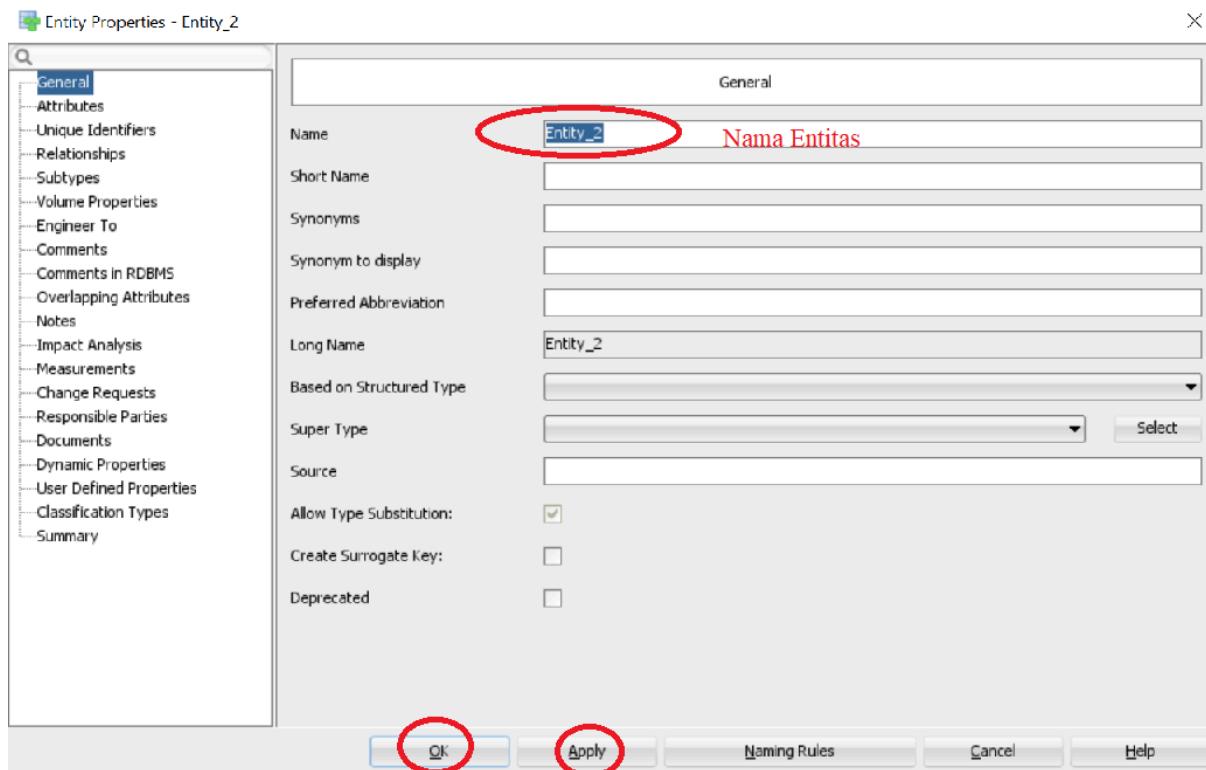
- Klik symbol new Entity pada bar menu



Kemudian bawa cursor ke layer Logical (Database1), buatlah gambar segi empat, sehingga menjadi seperti berikut:

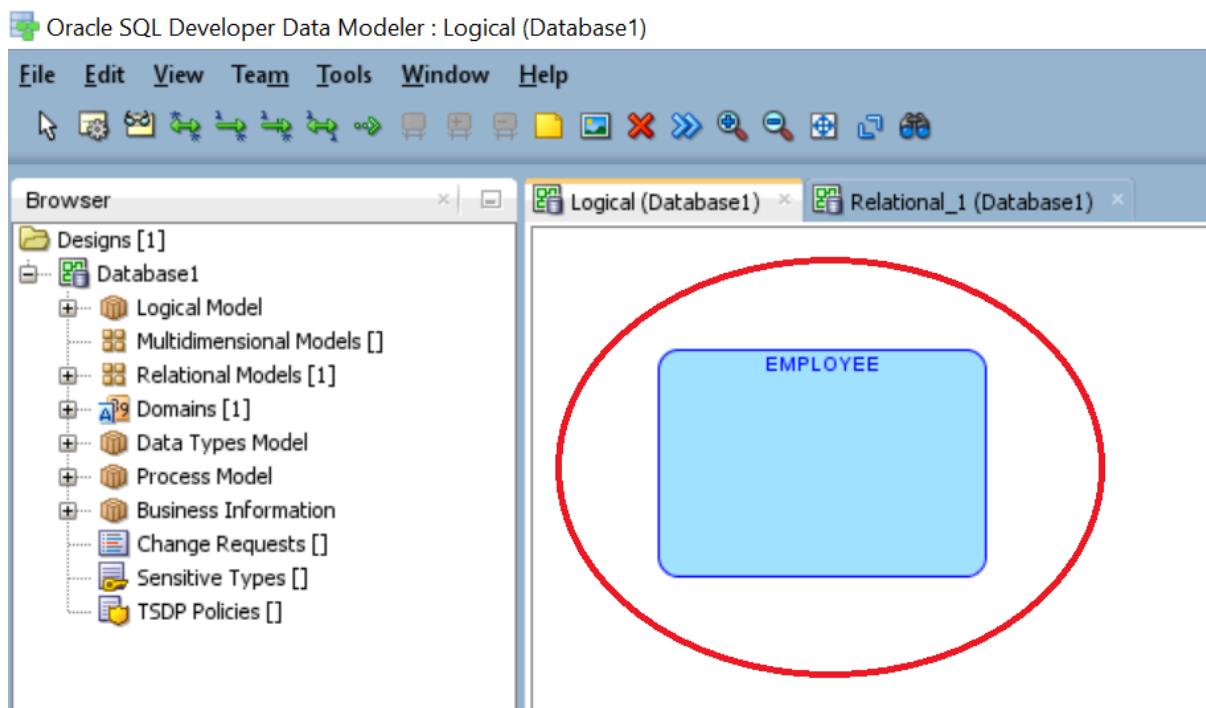


Setelah itu akan muncul layer Property dari entitas



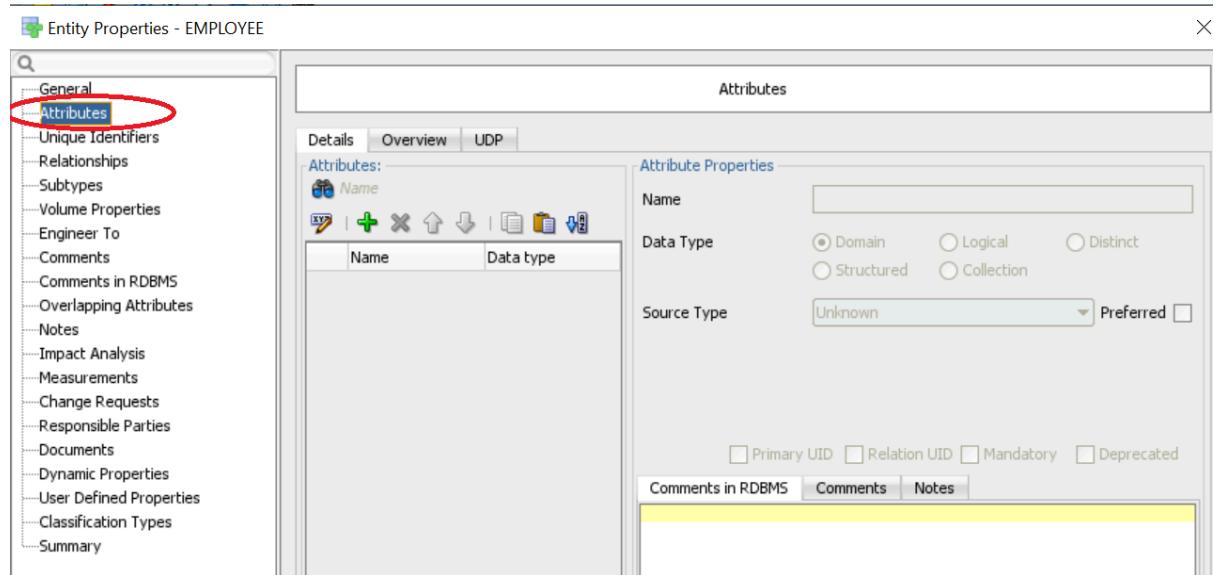
Beri nama entitas pada field Name, misalkan EMPLOYEE

Klik Apply, kemudian Ok, maka akan terbentuk satu entitas EMPLOYEE seperti berikut:

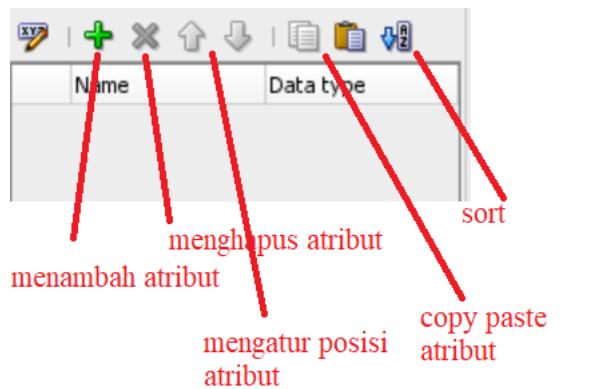


5. Buatlah Entitas yang lain dengan cara yang sama

6. Untuk menambahkan atribut pada Entitas, double klik pada entitas yang akan ditambahkan atribut, klik menu atribut (di menu sebelah kiri), maka akan muncul jendela atribut seperti berikut:



Anda bisa menambah, menghapus, mengubah atribut dengan meng-klik salah satu fungsi:



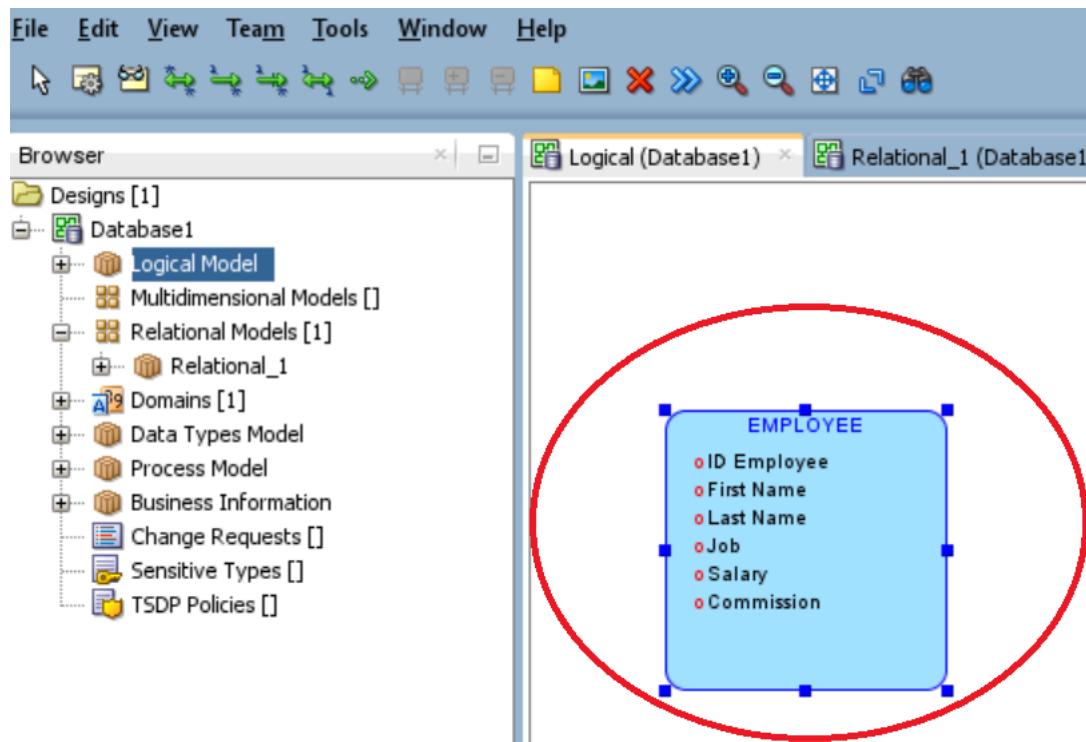
7. Beri Nama Atribut:

Attribute Properties	
Name	<input type="text" value="Attribute_1"/> Nama Atribut
Data Type	<input checked="" type="radio"/> Domain <input type="radio"/> Logical <input type="radio"/> Distinct <input type="radio"/> Structured <input type="radio"/> Collection
Source Type	<input type="text" value="Unknown"/> <input type="checkbox"/> Preferred

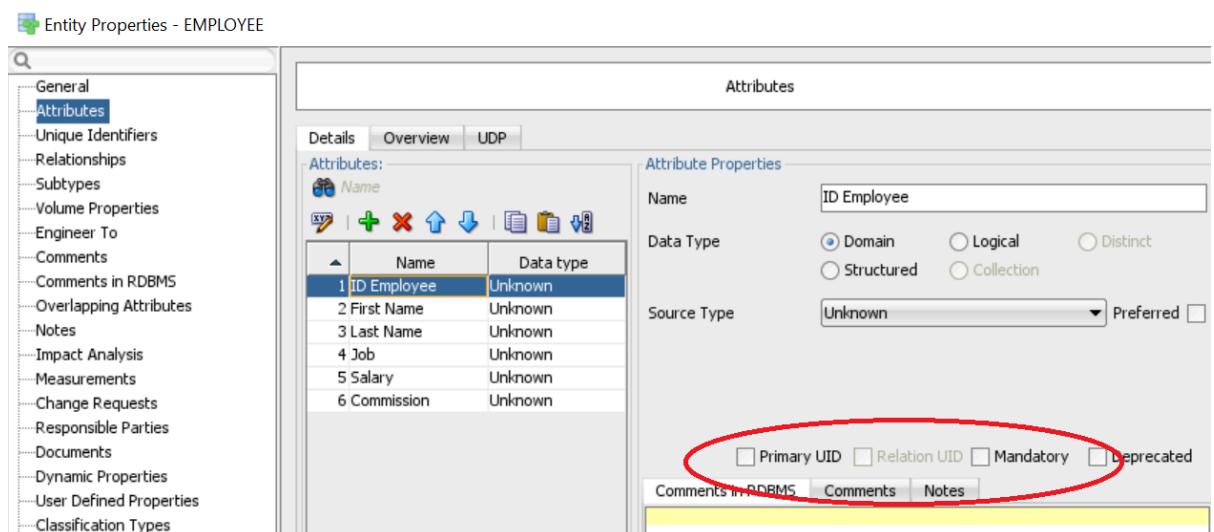
Misalkan pada entitas EMPLOYEE terdapat atribut: ID Employee, First Name, Last Name, Job, Salary, Commision.

Caranya:

Klik  , beri nama pada Atribut. Lakukan yang sama untuk semua atribut. Sehingga akan didapatkan seperti gambar berikut:



8. Untuk mengubah optionality pada atribut (mandatory/optional), di jendela Atribut Properties, check/uncheck Primary UID, atau Mandatory
- Defaultnya adalah Optional



Contoh:

Misalkan property dari atribut:

ID Employee : Primary UID

First Name, Last Name, Job, Salary : Mandatory

Commission : Optional

Details Overview UDP

Attributes:

Name	Data type
1 ID Employee	Unknown
2 First Name	Unknown
3 Last Name	Unknown
4 Job	Unknown
5 Salary	Unknown
6 Commission	Unknown

Attribute Properties

Name: ID Employee

Data Type: Domain

Source Type: Unknown

Buttons at the bottom: Primary UID (checked), Relation UID, Mandatory (checked), Deprecated

Attributes:

Name	Data type
1 ID Employee	Unknown
2 First Name	Unknown
3 Last Name	Unknown
4 Job	Unknown
5 Salary	Unknown
6 Commission	Unknown

Attribute Properties

Name: First Name

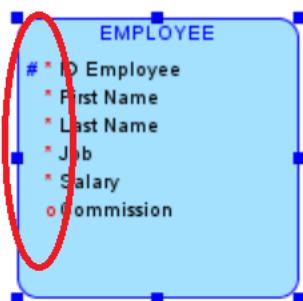
Data Type: Domain

Source Type: Unknown

Buttons at the bottom: Primary UID (unchecked), Relation UID, Mandatory (checked), Deprecated

Klik Apply kemudian OK

Sehingga optionality entity akan berubah menjadi sebagai berikut:



Keterangan Simbol di depan nama entitas

#* : Primary UID

● : Mandatory

○ : optional

TUGAS:

Berdasarkan masalah/business scenario sesuai kelompok masing-masing, gambarkan semua entitas dan atribut beserta optionality nya.

MODUL 2

RELASI

Tujuan Instruksional Umum

Mahasiswa mampu mendeskripsikan, dan menentukan relasi antar entitas

Tujuan Instruksional Khusus

1. Mahasiswa mampu memahami dan menginterpretasikan mengenai relasi
2. Mahasiswa mampu untuk mengidentifikasi hubungan antar entitas
3. Mahasiswa mampu mendeskripsikan kardinalitas dari suatu relasi
4. Mahasiswa mampu mengimplementasikan hubungan antar entitas dengan menerapkan aturan aturan kardinalitas.

MATERI PRAKTIKUM

1. RELASI

Relasi adalah suatu hubungan antara beberapa entitas. Setiap relasi mempunyai batasan (*constraint*) terhadap kemungkinan kombinasi entitas yang berpartisipasi. Batasan tersebut ditentukan dari situasi yang diwakili relasi tersebut.

Simbol relasi adalah:



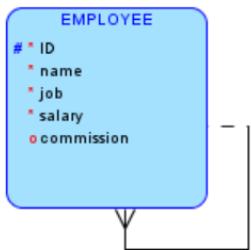
Derajat relasi:

1. Relasi *Unary / Recursive Relationship*.

Relasi unary/ rekursif adalah relasi yang menghubungkan 1 entity ke entitas itu sendiri.

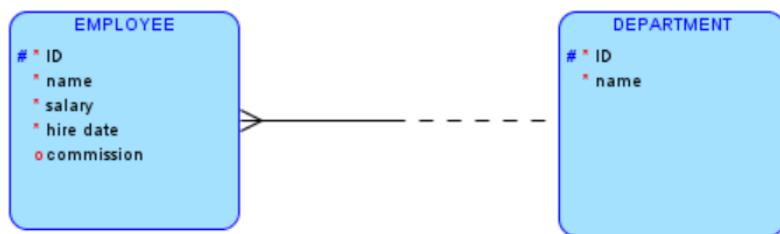
Contoh: Manager Penjualan yang mengelola para karyawannya yang merupakan penjual/salesman. Manager juga merupakan seorang karyawan.

Gambar Relasinya adalah:

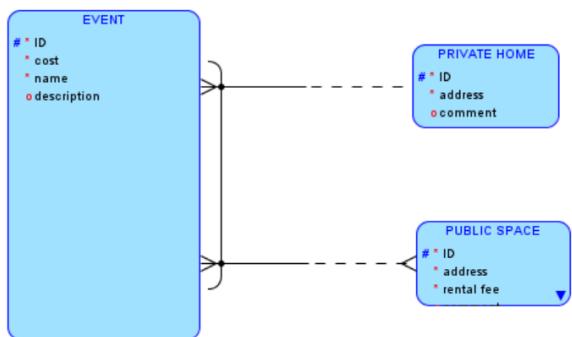
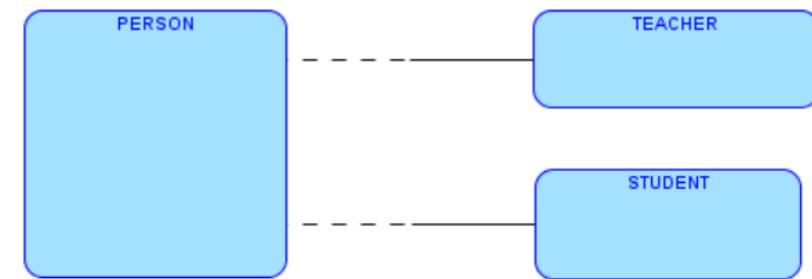


2. Relasi *Binary*. Relasi binary merupakan relasi antara dua entitas.

Contoh:



3. Relasi *Ternary*. Relasi ternary adalah merupakan relasi antara tiga entitas atau lebih.



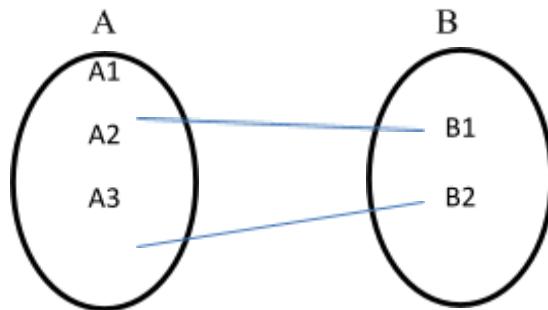
2. KARDINALITAS

Kardinalitas adalah jumlah entitas yang dapat berelasi dengan entitas pada himpunan entitas yang lain.

Terdapat 3 (tiga) macam Kardinalitas, yaitu :

- 1) Relasi One-to-one (notasi 1:1) :

Definisi: satu entitas di himpunan entitas A dapat berelasi dengan paling banyak 1 entitas di himpunan entitas B.

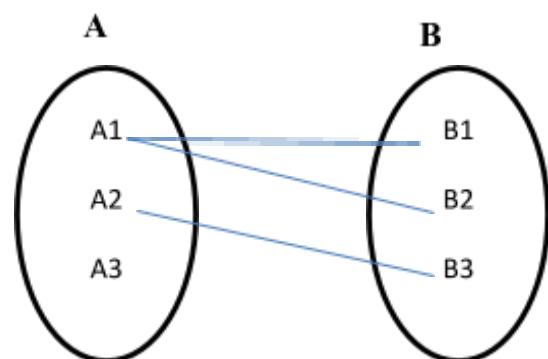


Contoh:

Relasi Pengemudi dan SIM: satu orang pengemudi hanya memiliki 1 SIM dan 1 SIM hanya dimiliki oleh 1 orang pengemudi

- 2) Relasi One-to-many (notasi 1:N) atau many-to-one (notasi N:1)

Definisi: 1 entitas di himpunan entitas A dapat berelasi dengan 1 atau lebih dari 1 entitas di himpunan B, dan 1 atau lebih dari 1 entitas di himpunan entitas B dapat berelasi dengan hanya 1 entitas di himpunan entitas A.

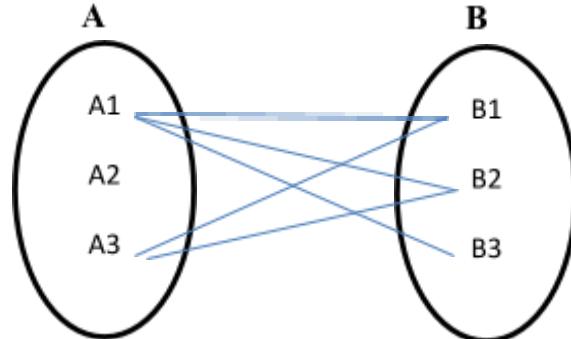


Contoh:

Suatu departemen di suatu perusahaan dapat memiliki lebih dari 1 karyawan, dan 1 karyawan hanya diperbolehkan berada di satu departemen.

3) Relasi Many-to-many (notasi M:N)

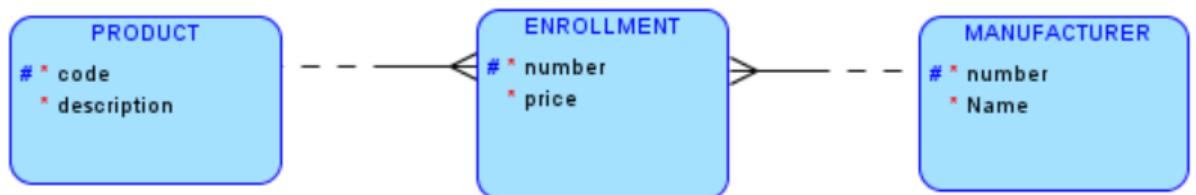
Definisi: 1 atau lebih dari 1 entitas di himpunan entitas A dapat berelasi dengan 1 atau lebih dari 1 entitas di himpunan B, dan 1 atau lebih dari 1 entitas di himpunan entitas B dapat berelasi dengan 1 atau lebih dari 1 entitas di himpunan entitas A.



Tipe-tipe Relasi

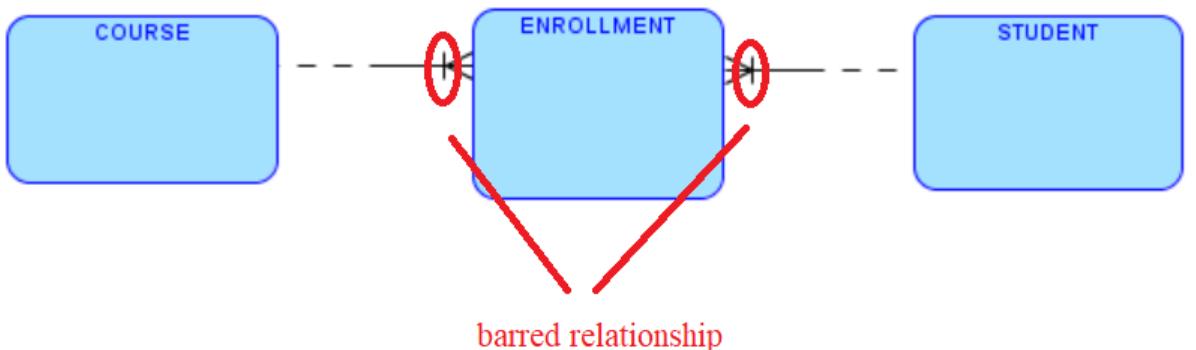
1. M:M relationship

Hubungan M:M akan diselesaikan dengan membuat suatu entitas baru, yaitu intersection entity, dan menjadi dua hubungan 1:M dan M:1.



2. Barred Relationship

UID dari intersection entity sering kali berasal dari relasi asal dan diwakili oleh bar. Dalam hal ini, hubungan dari entitas asal ke entitas persimpangan disebut barred relationship.

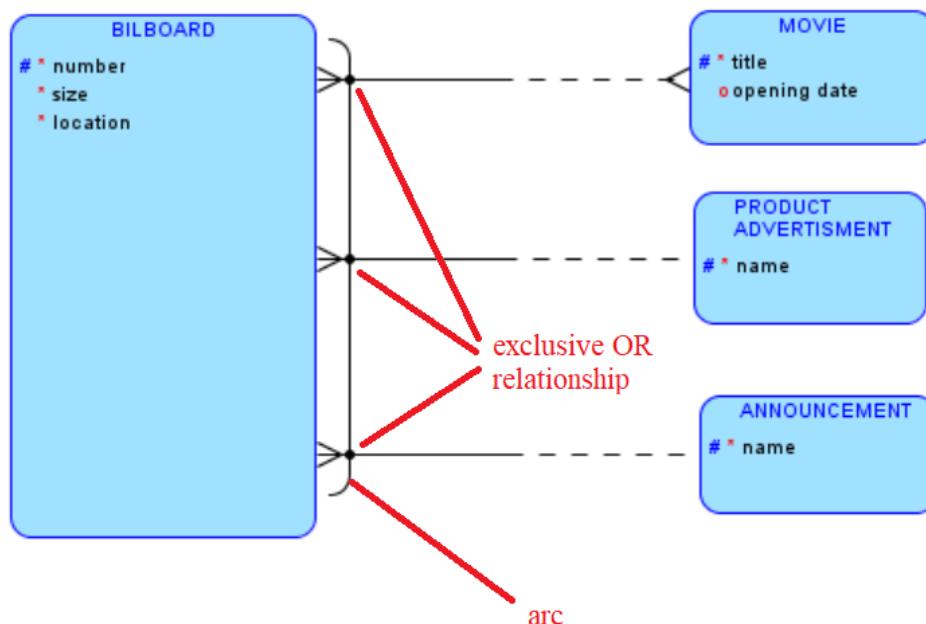


3. Arc Relationship

Arcs Relationship adalah cara untuk mewakili hubungan yang saling eksklusif di ERD dan diwakili pada ERD sebagai garis padat dengan ujung melengkung. Pada sebuah lingkaran digambar pada arc untuk setiap hubungan yang merupakan bagian dari arc.

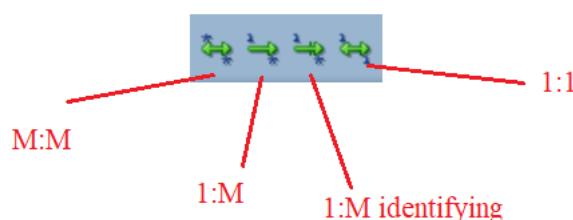
Karakteristik:

- Arcs selalu menjadi milik satu entitas
- Arcs dapat mencakup lebih dari dua hubungan.
- Tidak semua hubungan suatu entitas perlu dimasukkan dalam arc.
- Entitas mungkin memiliki beberapa arc
- Arc harus selalu terdiri dari hubungan dengan opsionalitas yang sama.
- Semua hubungan dalam arc harus bersifat wajib atau semua harus opsional.
- Hubungan dalam arc mungkin berbeda kardinalitas, meskipun ini jarang terjadi.

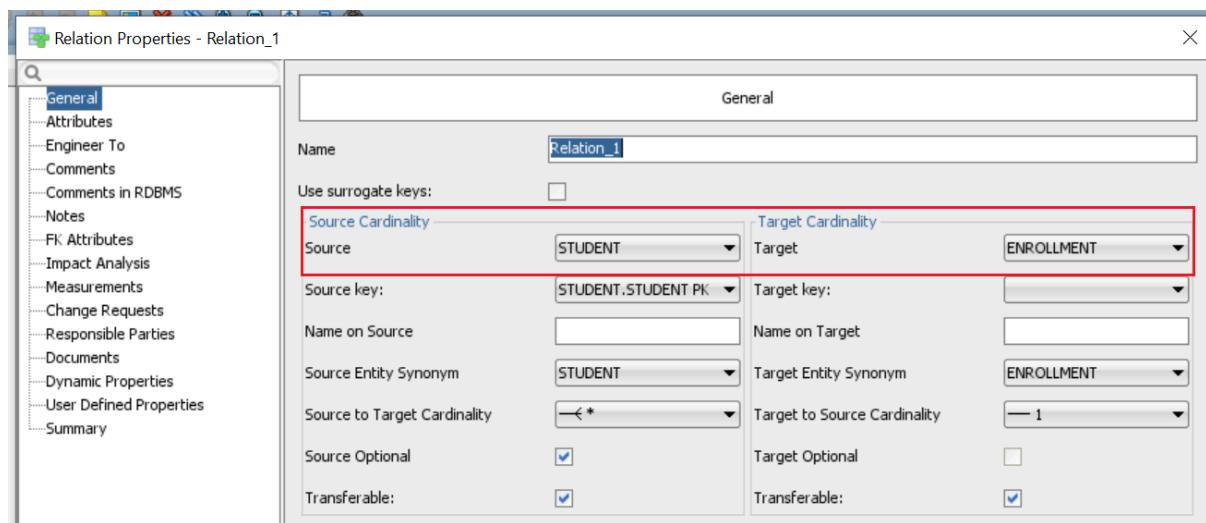


Untuk membuat Relasi di data modeler:

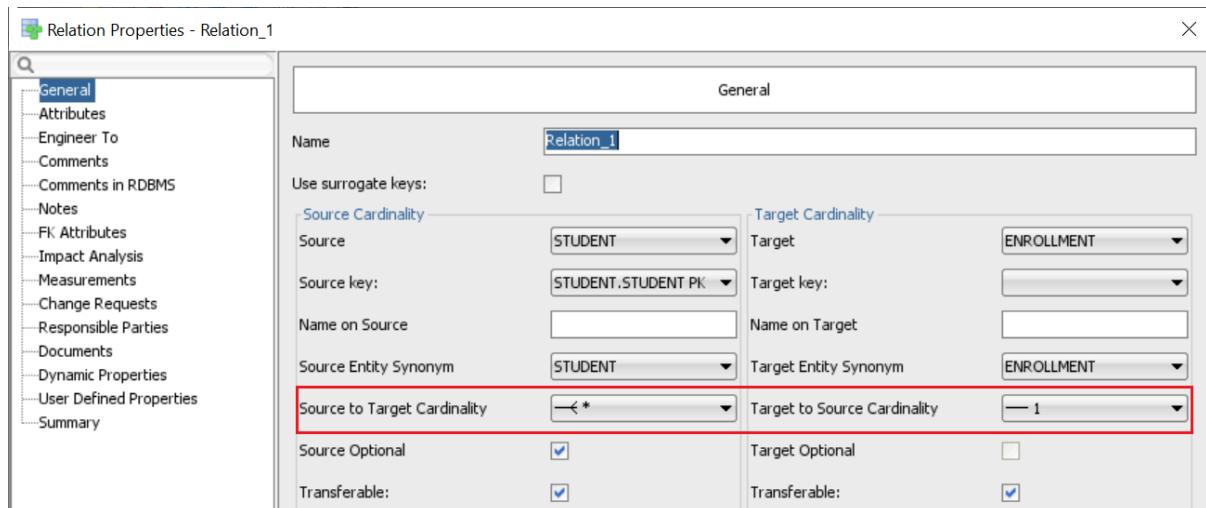
1. Tentukan entitas yang akan di relaskan
2. Tentukan kardinalitas dan optionality dari relasi



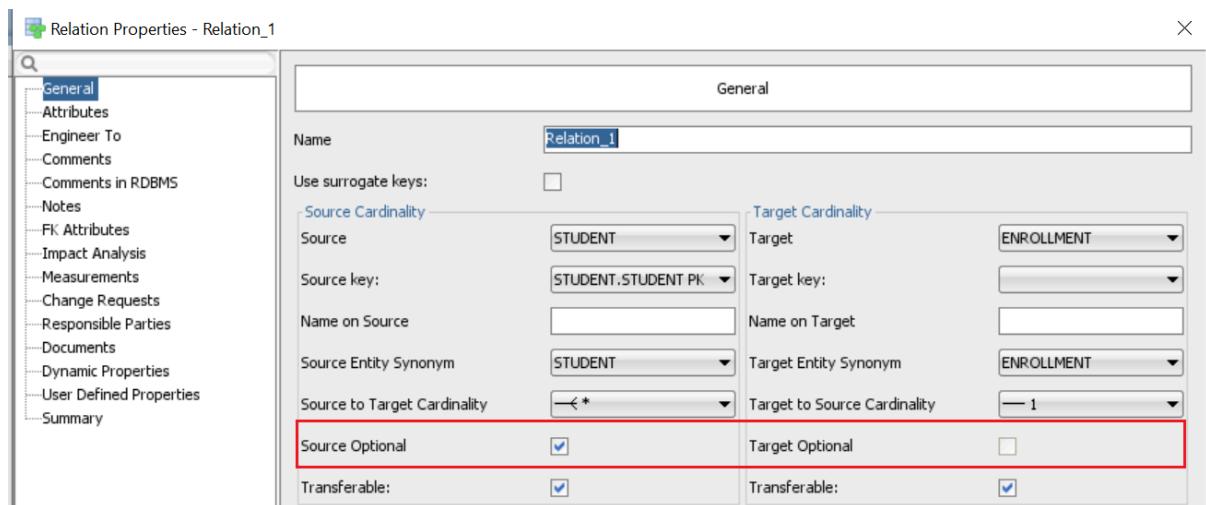
3. Klik di entitas yang akan di relaskan. Entitas pertama yang akan direlasikan disebut source, dan entitas yang dihubungkan disebut target.



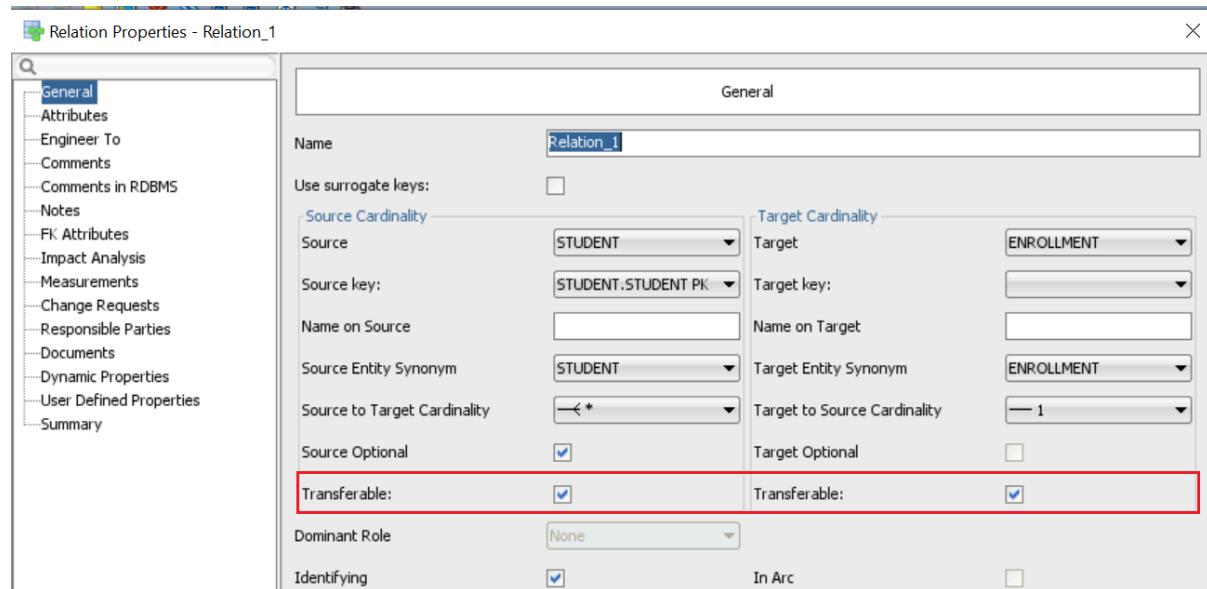
Anda bisa mengubah kardinalitas dari relationship di sini:



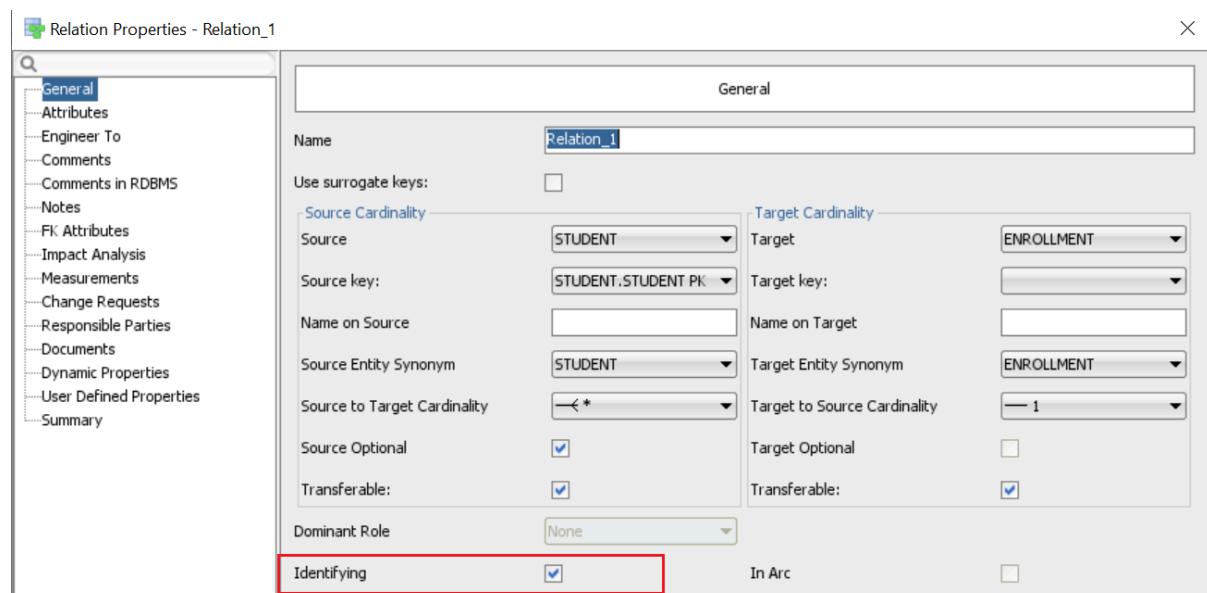
- Untuk mengubah optionality, check/ uncheck menu optionality sebagai berikut: (default adalah optional, jika relasi mandatori un-check optional nya)



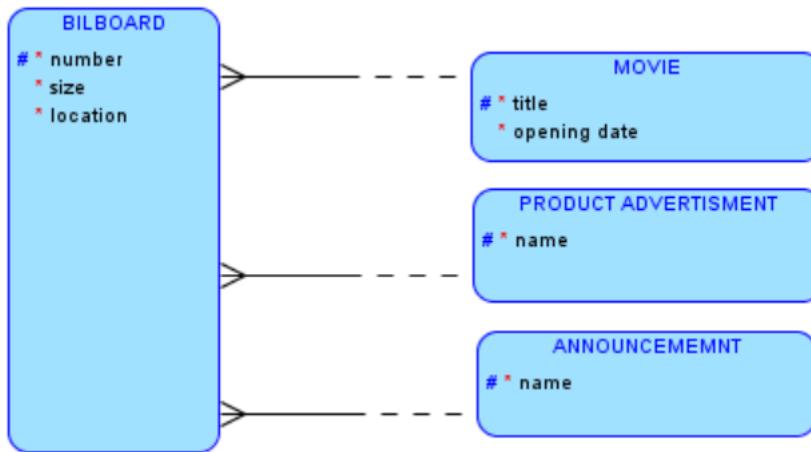
5. Untuk membuat relasi transferable dan untransferable, check/ uncheck menu transferable (default : transferable)



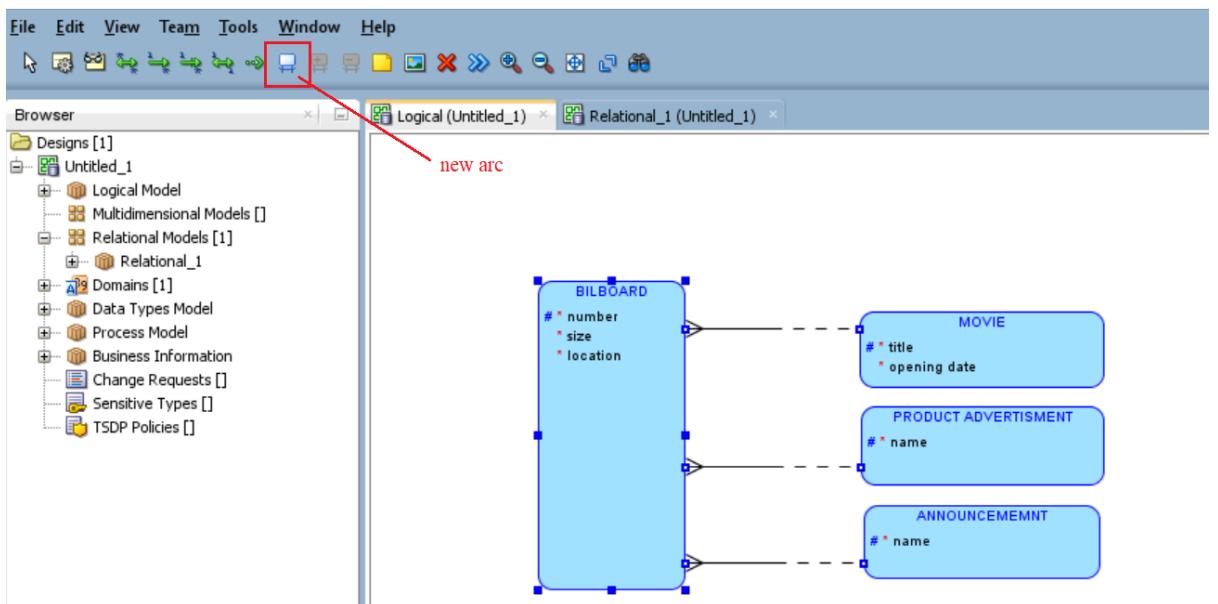
6. Untuk membuat barred relationship, check/ uncheck menu identifying berikut:



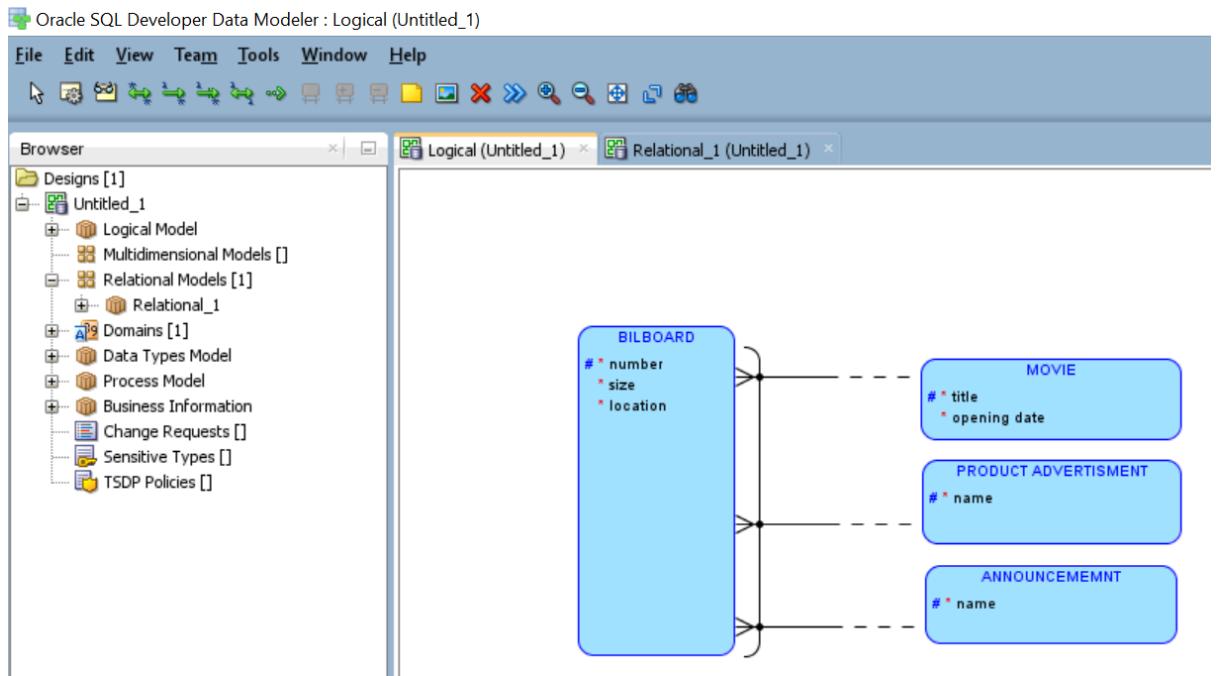
7. Untuk membuat arc relationship, misal contoh berikut ini:



1. Klik BILLBOARD (supertype),
2. Tekan ctrl dan klik di semua relasi yang termasuk dalam arc relationship (dalam contoh ini semua relasi di gambar akan masuk di arc relationship) dan tombol new arc akan muncul



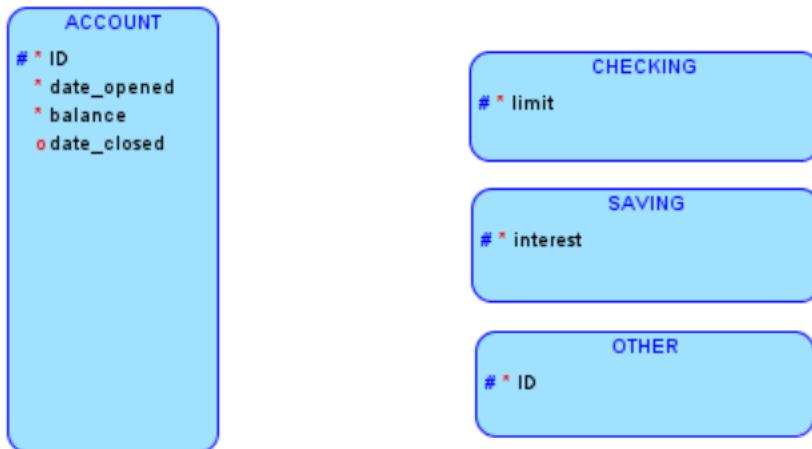
3. Klik tombol new arc



SUPERTYPE DAN SUBTYPE

Untuk membuat supertype dan subtype:

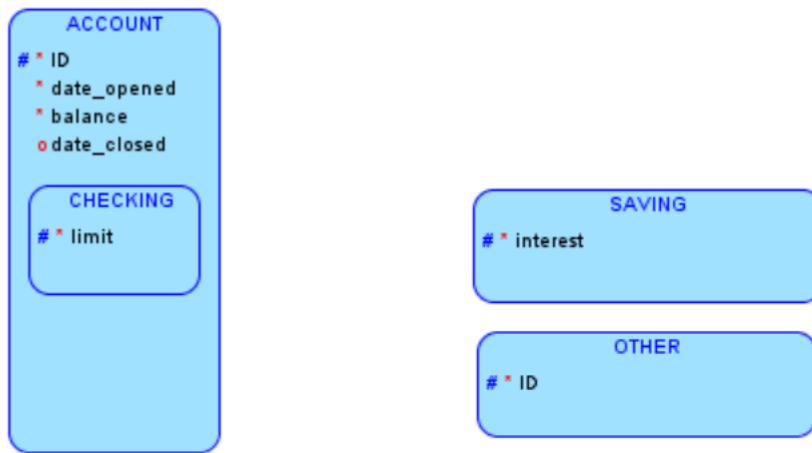
Sebagai contoh:



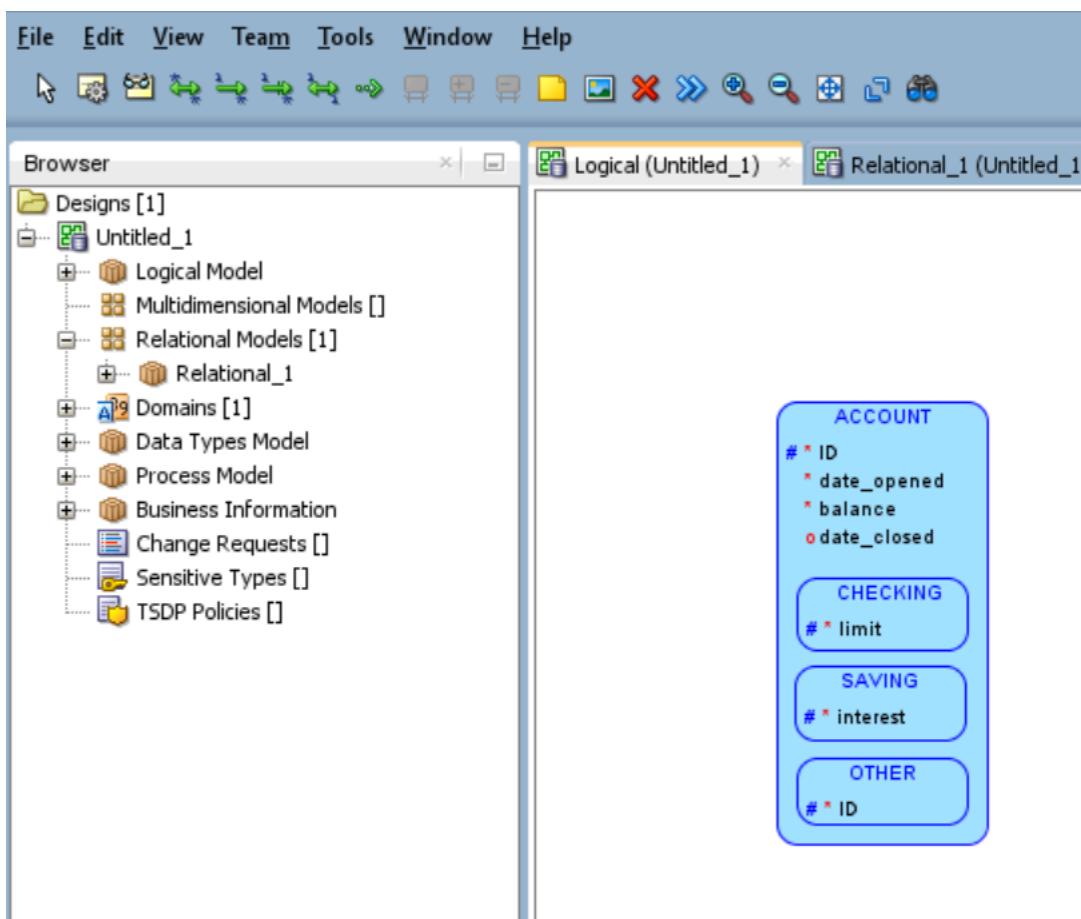
ACCOUNT merupakan supertype, sedangkan CHECKING, SAVING, OTHER merupakan subtype.

Untuk membuat subtype berada di dalam supertype:

1. Klik entitas CHECKING, kemudian pilih menu supertypem akan muncul dropdown berisi nama semua entitas yang ada. Pilih entitas yang menjadi supertype nya, yaitu ACCOUNT.
2. Secara otomatis, entitas CHECKING akan berada di dalam supertype nya.



3. Langkah yang sama untuk semua subtype.



Entity Properties - CHECKING

General

Name	CHECKING
Short Name	
Synonyms	
Synonym to display	
Preferred Abbreviation	
Long Name	CHECKING
Based on Structured Type	
Super Type	ACCOUNT
Source	OTHER
Allow Type Substitution:	SAVING
Create Surrogate Key:	<input type="checkbox"/>

8. Membuat supertype sub type:

MODUL 3

NORMALISASI

Tujuan Instruksional Umum

Mahasiswa mampu membangun desain logika basis data relasional untuk menghasilkan struktur tabel yang normal

Tujuan Instruksional Khusus

1. Mahasiswa mampu mengidentifikasi tabel bentuk normal 1, bentuk normal 2 dan bentuk normal 3.
2. Mahasiswa mampu melakukan proses normalisasi

MATERI PRAKTIKUM

1. NORMALISASI

Normalisasi merupakan Teknik/ pendekatan yang digunakan dalam membangun design logic database relasional melalui organisasi himpunan data dengan tingkat ketergantungan fungsional dan keterkaitan yang tinggi sedemikian sehingga menghasilkan struktur tabel yang normal.

Tujuan Normalisasi adalah:

- a. Minimalisasi redundansi (pengulangan data)
- b. Memudahkan identifikasi entitas
- c. Mencegah terjadinya anomali

Beberapa bentuk normal (normal forms, NF) :

- 1NF, 2NF, 3NF, BCNF: based on keys and functional dependencies
- 4NF, 5NF: based on keys and multivalued dependencies

2. FIRST NORMAL FORM (1NF)

Suatu relasi disebut memenuhi bentuk normal pertama (1NF) jika dan hanya jika setiap atribut dari relasi tersebut hanya memiliki nilai tunggal dan tidak ada pengulangan grup atribut dalam baris.

Contoh 1:

Tabel: Sales

IDSales	NamaSales	Telepon
ADN006	Yeni, SE	3517261, 3520165
ADN007	Memey	4744621, 08122861427
ADN008	Tina	08566241521
ADN009	Ir. Yanto	7265122, 7123910
ADN010	Made	6723192

IDSales	NamaSales	Telepon
ADN006	Yeni, SE	3517261
ADN006	Yeni, SE	3520165
ADN007	Memey	4744621
ADN007	Memey	08122861427
ADN008	Tina	08566241521
ADN009	Ir. Yanto	7265122
ADN009	Ir. Yanto	7123910
ADN010	Made	6723192

Contoh 2:

Tabel: Buku

ISBN	Thn_Terbit	ID_Pengarang	Nama_Pengarang	ID_Pengarang	Nama_Pengarang
12-1202-19222	1992	K0121	Aris M	K1021	Kosim P
11-1090-29101	2001	K1021	Kosim P		
11-1090-29102	2001	K2091	K Odelia	K0121	Aris M
12-1201-90871	2002	K2092	Renaldi	K2091	K Odelia
13-2089-12910	2001	K2019	Samsuri J		

ISBN	Thn_Terbit	ID_Pengarang	Nama_Pengarang
12-1202-19222	1992	K0121	Aris M
12-1202-19222	1992	K1021	Kosim P
11-1090-29101	2001	K1021	Kosim P
11-1090-29102	2001	K2091	K Odelia
11-1090-29102	2001	K0121	Aris M
12-1201-90871	2002	K2092	Renaldi
12-1201-90871	2002	K2091	K Odelia
13-2089-12910	2001	K2019	Samsuri J

3. SECOND NORMAL FORM (2NF)

Suatu relasi disebut memenuhi bentuk normal kedua (2NF) jika dan hanya jika:

1. Memenuhi 1NF
2. setiap atribut yang bukan kunci utama (primary key) tergantung secara fungsional terhadap semua atribut kunci dan bukan hanya sebagian atribut kunci (fully functionally dependent).

Untuk normalisasi ke bentuk 2NF, maka tabel 1NF didekomposisi menjadi beberapa tabel yang masing-masing memenuhi 2NF.

Contoh 1:

Diketahui tabel R = (A,B,C,D,E) ; A,B kunci utama (primary key)

Dengan FD : A,B \square C,D,E

Maka tabel R memenuhi 2NF sebab:

A,B \square C,D,E berarti:

A,B \square C,

A,B \square D dan

A,B \square E

Jadi semua atribut bukan kunci utama tergantung penuh pada (A,B).

Contoh 2:

Bagaimana bila R = (A,B,C,D,E) tetapi dengan FD : (A,B) \square (C,D) dan B \square E. Apakah memenuhi 2NF ?

Jelas bahwa R bukan 2NF karena ada atribut E yang bergantung hanya pada atribut B saja dan bukan terhadap (A,B).

Dari FD : (A,B) \square (C,D) juga mencerminkan bahwa hanya C dan D saja yang bergantung secara fungsional terhadap (A,B), tidak untuk E.

Jadi bukan 2NF.

Untuk mengubah menjadi 2NF, lakukan dekomposisi menjadi:

R1= (A,B,C,D) dan R2= (B,E).

R1 dan R2 memenuhi 2NF.

Contoh 3:

Diketahui suatu entitas Workshop = (NIM, Modul, Biaya, Grade)

NIM	Modul	Biaya	Grade
P11.2004.0129	VB.Net	250000	A

Key : (NIM, Modul)

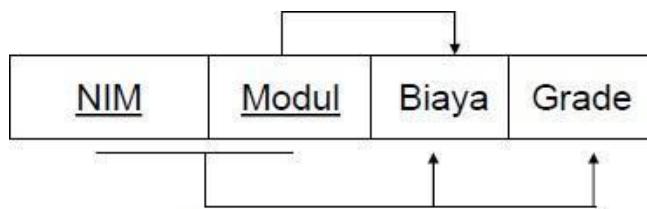
FD : Modul \square Biaya

(Biaya ditentukan oleh Modul yang diambil mahasiswa) : sudah memenuhi bentuk 1NF tetapi tidak 2NF

Tabel biaya peserta workshop

NIM	Modul	Biaya	Grade
P11.2004.0129	VB.Net	250000	A
P11.2004.0130	Prolog	100000	A
P11.2004.0129	Prolog	100000	B
P11.2004.0201	Delphi 6	150000	A
P11.2004.0250	VB.Net	250000	B

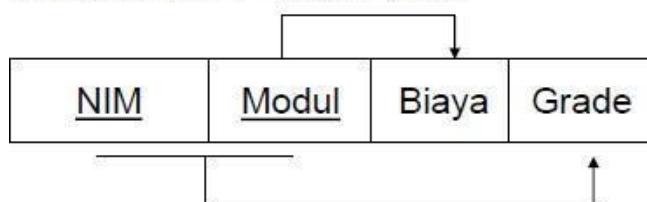
Sebab dalam tabel ini, biaya tidak bergantung penuh pada atribut kunci (NIM, Modul)



$(NIM, Modul) = \text{key}$

$(NIM, Modul) \rightarrow Biaya$ (partial) ← Eliminate

$(NIM, Modul) \rightarrow Grade$ (full)



Make Decomposition :

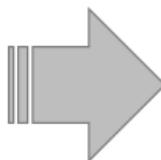
Works1 = $(NIM, Modul, Grade)$

Works2 = $(Modul, Biaya)$

Fully Dependency

Workshop

NIM	Modul	Biaya	Grade
P11.2004.0129	VB.Net	250000	A
P11.2004.0130	Prolog	100000	A
P11.2004.0129	Prolog	100000	B
P11.2004.0201	Delphi 6	150000	A
P11.2004.0250	VB.Net	250000	B



NIM	Modul	Grade
P11.2004.0129	VB.Net	A
P11.2004.0130	Prolog	A
P11.2004.0129	Prolog	B
P11.2004.0201	Delphi 6	A
P11.2004.0250	VB.Net	B

Works1

Modul	Biaya
VB.Net	250000
Prolog	100000
Delphi 6	150000

Works2

4. THIRD NORMAL FORM (3NF)

Suatu relasi disebut memenuhi bentuk normal ketiga (3NF) jika dan hanya jika:

- memenuhi 2NF

Key (tidak terdapat ketergantungan transitif pada atribut bukan kunci).

Atau

Suatu relasi disebut memenuhi bentuk normal ketiga (3NF) jika dan hanya jika setiap FD nontrivial : $X \rightarrow A$, di mana X dan A atribut (atau kompositnya), memenuhi salah satu kondisi:

1. X adalah superkey
2. A merupakan anggota candidate key (A disebut prime attribute)

Jika suatu relasi sudah memenuhi 2NF tapi tidak memenuhi 3NF, maka untuk normalisasi ke bentuk 3NF, tabel 2NF didekomposisi menjadi beberapa tabel hingga masing-masing memenuhi 3NF.

Catatan:

Jika suatu relasi memenuhi 2NF dan hanya memiliki tepat satu atribut yang bukan kunci utama maka relasi tersebut sudah memenuhi 3NF

Contoh:

Diketahui tabel R = (A,B,C,D,E) ; A, B kunci utama(primary key) dengan FD : A,B \rightarrow C,D,E dan C \rightarrow D,E maka R bukan 3NF sebab: Atribut D dan E (bukan kunci utama) bergantung secara fungsional pada C (yang juga bukan kunci utama).

Melalui FD :

Diketahui A,B \rightarrow C,D,E.

Karena sifat refleksif maka A,B \rightarrow A,B. Sehingga A,B \rightarrow A,B,C,D,E

(A,B) : Superkey.

Diketahui C \rightarrow D,E.

Karena sifat refleksif maka C \rightarrow C. Sehingga C \rightarrow C,D,E.

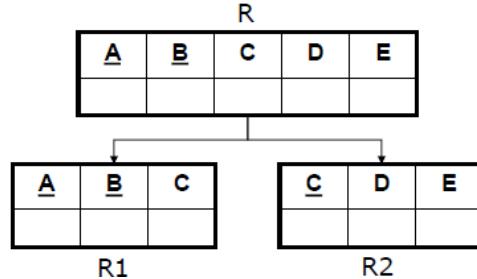
Karena C \rightarrow A,B,C,D,E maka C bukan superkey.

Jadi: Tidak memenuhi definisi 3NF. Jadi R bukan 3NF.

Agar R memenuhi 3NF maka didekomposisi menjadi: R1=(A,B,C) dan R2=(C,D,E) sehingga R1 dan R2 memenuhi 3NF.

FD : $A, B \rightarrow C, D, E$ berarti $A, B \rightarrow C$; $C \rightarrow D, E$; $A, B \rightarrow D, E$
 $A, B \rightarrow D$ reduce }
 $A, B \rightarrow E$ reduce }

Dekomposisinya : $R1 = (A, B, C)$; FD : $(A, B) \rightarrow C$
 $R2 = (C, D, E)$; FD : $C \rightarrow D, E$



Contoh:

Misal diketahui struktur informasi dari suatu dokumen supplier :

S	Status	City	PQ	
			P	Qty
S1	20	LONDON	P1	300
			P2	200
			P3	400
			P4	200
			P5	100
			P6	100
S2	10	PARIS	P1	300
			P2	400
S3	10	PARIS	P2	200
S4	20	LONDON	P2	200
			P4	399
			P5	400

Akan dibentuk suatu tabel dengan skema

TPS = (S, Status, City, P, Qty)

Dengan (S,P) = primary key

Dan berlaku FD :

S \square Status

S \square City

City \square Status

Lakukan normalisasi dari 1NF hingga 3NF.

JAWAB:

TPS

S	Status	City	P	Qty
S1	20	LONDON	P1	300
S1	20	LONDON	P2	200
S1	20	LONDON	P3	400
S1	20	LONDON	P4	200
S1	20	LONDON	P5	100
S1	20	LONDON	P6	100
S2	10	PARIS	P1	300
S2	10	PARIS	P2	400
S3	10	PARIS	P2	200
S4	20	LONDON	P2	200
S4	20	LONDON	P4	399
S4	20	LONDON	P5	400

- 1NF

- Not 2NF Problem :

Redundansi \square inconsistency (low speed process)

- Anomaly :

S \square (Status, City) tapi tidak bisa dilakukan insert data (S5, 30, JAKARTA) tanpa diikuti data P (khususnya) dan Qty.

Menghapus 1 baris data akan merusak keutuhan informasi.

Solusi:

Dekomposisi menjadi:

TPS1 dan TPS2

TPS1

S	Status	City
S1	20	LONDON
S2	10	PARIS
S3	10	PARIS
S4	20	LONDON

- 1NF
- 2NF
- Not 3NF (trans.)
S → City
City → Status

- Sekarang kita dapat menambah data (S5,30,JAKARTA) secara aman
- Tapi masih ada anomaly :
Karena **City → Status** maka kita tidak bisa entry data City baru sebelum Status punya nilai. Penghapusan 1 baris sebagian data City juga bisa merusak keutuhan informasi S.
- Selain itu, masih ada redundansi pada Status dan City

TPS2

S	P	Qty
S1	P1	300
S1	P2	200
S1	P3	400
S1	P4	200
S1	P5	100
S1	P6	100
S2	P1	300
S2	P2	400
S3	P2	200
S4	P2	200
S4	P4	399
S4	P5	400

- 1NF
- 2NF
- 3NF

Redundansi partial
→ tidak potensial
→ lebih baik dari redundan sebelumnya

Tidak mungkin menghilangkan semua redundan tapi buat yang minimal

TPS1-1

S	City
S1	LONDON
S2	PARIS
S3	PARIS
S4	LONDON

- 1NF
- 2NF
- 3NF

TPS1-2

City	Status
LONDON	20
PARIS	10

- 1NF
- 2NF
- 3NF

TPS2

S	P	Qty
S1	P1	300
S1	P2	200
S1	P3	400
S1	P4	200
S1	P5	100
S1	P6	100
S2	P1	300
S2	P2	400
S3	P2	200
S4	P2	200
S4	P4	399
S4	P5	400

- 1NF
- 2NF
- 3NF

--	--	--

Praktikum #5

Dasar-Dasar Pengoperasian MySQL

Standar Kompetensi

Mahasiswa dapat mengoperasikan MySQL untuk membuat basis data dan tabel-tabel untuk suatu basis data.

Kompetensi Dasar	Indikator
1. Mampu mengoperasikan MySQL	a. Mengetahui cara membuka MySQL. b. Mengetahui perintah dasar MySQL. c. Mampu melakukan <i>setting</i> tampilan dan warna huruf, warna latar belakang, serta ukuran windows MySQL. d. Mengetahui cara menutup MySQL.
2. Mampu membuat basis data	a. Mengetahui cara menampilkan basis data yang sudah ada. a. Mampu membuat basis data baru. b. Mampu menghapus basis data.
3. Mampu membuat tabel	a. Mampu membuat tabel baru. b. Mampu melengkapi tabel dengan kolom. c. Mengetahui cara menampilkan kolom-kolom dari suatu tabel.
4. Mampu meremajakan tabel	a. Mampu menambah kolom baru. b. Mampu mengganti nama kolom. c. Mampu mengganti lebar dan jenis kolom. d. Mampu mengganti nama tabel. e. Mampu menghapus tabel.

A. Basis data dan Sistem Manajemen Basis Data (SMBD)

Basis data (*database*) merupakan sekumpulan data yang saling berhubungan, dengan minimum redundancy. Untuk mengelola basis data digunakan perangkat lunak yang disebut dengan Sistem Manajemen Basis Data. Salah satu bentuk SMBD yang sekarang banyak digunakan ialah SMBD Relational.

Di dalam SMBD Relasional semua data disimpan dalam sekumpulan tabel, di mana sebuah tabel menyimpan informasi mengenai sebuah objek tertentu. Dengan SMBD Relasional, sebuah basis data akan mudah dikelola walaupun memiliki jumlah data yang banyak dan kompleks.

Pada prinsipnya sebuah SMBD Relasional terdiri atas tiga bagian, ialah:

1. Data Definition

Mendefinisikan jenis data yang akan dibuat (seperti berupa angka atau huruf), cara relasi data, validasi data, dan lainnya.

2. Data Manipulation

Proses manipulasi terhadap data yang telah dibuat, seperti menyaring data, melakukan proses query, dan sebagainya.

--	--	--

3. Data Control

Bagian ini berhubungan dengan cara mengendalikan data, seperti siapa saja yang bias melihat isi data, bagaimana data dapat digunakan, dan sebagainya.

MySQL (baca: mai es kju el) termasuk jenis SMBD Relasional yang dikembangkan oleh perusahaan Swedia bernama MySQL AB. MySQL dapat dijalankan dengan menggunakan sistem operasi yang sangat popular saat ini, yaitu Windows atau Linux.

Sejak Versi 3.23.19 MySQL merupakan perangkat lunak basis data yang bebas berlisensi GPL (*General Public License*), yang artinya: “*Source code MySQL dapat dilihat dan gratis, serta server MySQL dapat dipakai tanpa beaya untuk kebutuhan apapun. Tetapi jika Anda memodifikasi source code, Anda juga harus melepasnya dibawah lisensi yang sama, yaitu GPL.*”

Sebagai suatu SMBD Relasional maka di dalam MySQL digunakan istilah-istilah tabel, baris dan kolom. Sebuah basis data mengandung sejumlah tabel, dan setiap tabel terdiri atas sejumlah baris dan kolom. Istilah lain yang digunakan untuk baris ialah **record**, dan untuk kolom ialah **field**.

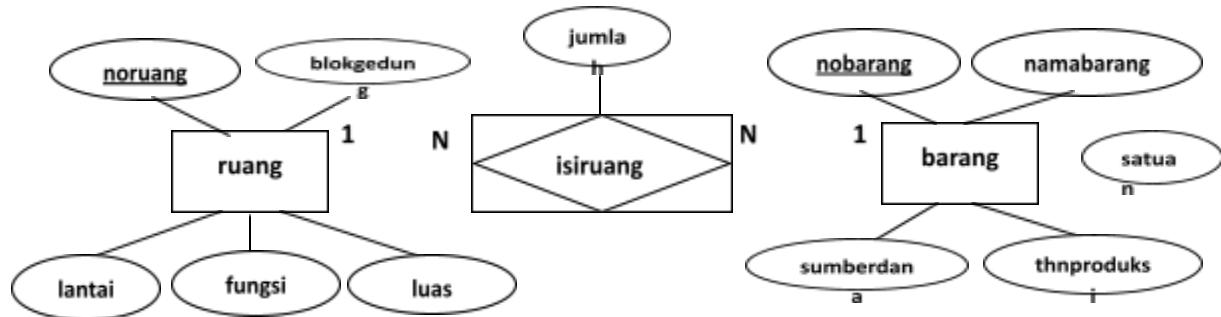
Contoh:

Data mengenai ruang dan barang suatu kantor disimpan dengan menggunakan schema relasi sebagai berikut:

ruang (noruang, blokgedung, lantai, fungsi, luas)
barang (nobarang, namabarang, sumberdana, thnproduksi, satuan)

Setiap ruang paling sedikit memiliki satu jenis barang dan di dalam suatu ruang dapat berisi lebih dari satu unit barang yang sama.

Diagram E/R dari basis data di atas ialah sebagai berikut:



Dengan kardinalitas *relationship* yang M-N, dari gambar di atas terlihat muncul atribut *relationship* atau atribut *intersection* untuk menyimpan data tentang barang yang ada di dalam ruang.

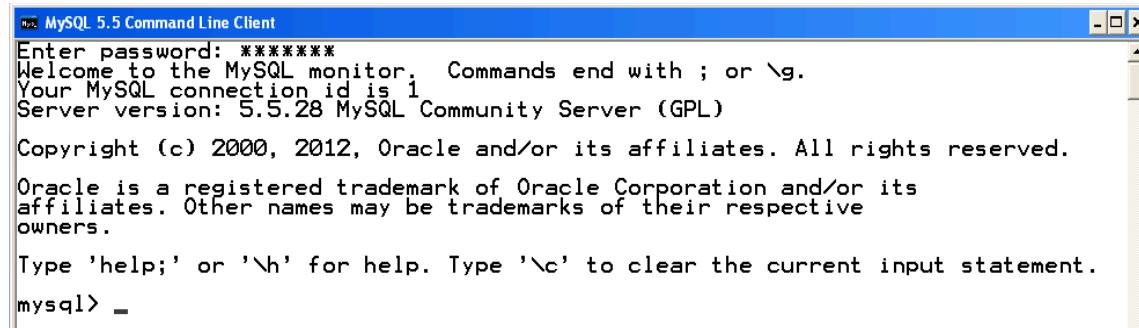
B. Pengoperasian MySQL

Untuk menjalankan MySQL Server dari menu utama Windows, lakukan klik:

Windows Start -> All Programs -> MySQL -> MySQL Server 5.5 -> MySQL Command Line Client

--	--	--

Masukkan *password* sesuai yang ditentukan, kemudian jika *password* dinyatakan valid, tampilan yang diberikan ialah sebagai berikut:



```
MySQL 5.5 Command Line Client
Enter password: *****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 1
Server version: 5.5.28 MySQL Community Server (GPL)

Copyright (c) 2000, 2012, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> _
```

Baris terakhir di atas berupa *prompt* yang merupakan tanda bahwa MySQL sudah siap menerima perintah selanjutnya, dan posisi kotak yang berkedip merupakan posisi di mana hasil pengetikan perintah akan ditampilkan.

Untuk melihat perintah-perintah yang disediakan MySQL ketik “**help;**” atau berikan “**/h**”.

Tampilan *prompt* dapat diganti sesuai keinginan, misal *prompt optional* berupa **mysql>** akan diganti dengan **prodi_stat>**, berikan perintah berikut berikan perintah:

```
mysql> prompt prodi_stat>
```

Lihat apa yang berubah pada tampilan prompt. Kemudian untuk mengembalikan ke tampilan prompt yang baku, ketikkan perintah “**prompt**”, yang akan mengembalikan tampilan prompt sebagai berikut:drop

```
mysql>
```

Apabila diinginkan *setting* warna dan ukuran huruf, ukuran windows, warna layar latar belakang, dan lain-lain, klik tombol kecil pada pojok kiri atas dari windows MySQL, kemudian pilih **Properties**.

Selanjutnya dari pilihan *setting* yang ditampilkan lakukan *setting* sesuai yang diinginkan, sampai diperoleh hasil yang paling sesuai.

Gunakan tombol panah ke atas atau ke bawah untuk menampilkan perintah-perintah baris yang sudah diberikan sebelumnya.

Pengoperasian MySQL diakhiri dengan mengetikkan “quit” atau “/q”, dan jika berhasil windows dari MySQL akan ditutup.

C. Membuat basis data

1. Melihat basis data yang sudah ada:

Untuk melihat basis data yang sudah ada digunakan perintah “**show databases**” dan diakhiri dengan “;” sebagai berikut:

--	--	--

```
mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| test |
+-----+
4 rows in set (0.47 sec)
```

Dari tampilan yang diberikan dapat dilihat basis data apa saja yang sudah disimpan di dalam sistem.

2. Pembuatan basis data baru

Misal basis data dari contoh di atas diberi nama **inventory**, maka untuk pembuatan basis data tersebut digunakan perintah “**create database**” sebagai berikut:

```
mysql> create database inventory;
Query OK, 1 row affected (0.38 sec)
```

Kemudian untuk cek hasilnya, berikan perintah “**show databases**” sebagai berikut:

```
mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| inventory |
| mysql |
| performance_schema |
| test |
+-----+
5 rows in set (0.00 sec)
```

Dari tampilan di atas ternyata basis data **inventory** sudah di-create oleh MySQL.

3. Penghapusan basis data

Untuk menghapus basis data **inventory**, perintah yang digunakan ialah “**drop database**”, sebagai berikut.

Cek sebelum penghapusan:

```
mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| inventory |
| mysql |
| performance_schema |
| test |
+-----+
5 rows in set (0.00 sec)
```

Perintah penghapusan:

```
mysql> drop database inventory;
```

Cek hasil penghapusan:

--	--	--

```
mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| test |
+-----+
4 rows in set (0.00 sec)
```

Perlu diperhatikan bahwa penghapusan suatu basis data memberikan efek pada terhapusnya semua tabel yang ada di dalamnya.

D. Pembuatan tabel

1. Membuat tabel baru

Sebelum pembuatan tabel, aktifkan dulu basis data dari tabel yang akan dibuat dengan perintah sebagai berikut:

```
mysql> use inventory;
Database changed
```

Kemudian untuk melakukan cek tabel-tabel yang sudah ada di dalam basis data tersebut, berikan perintah sebagai berikut:

```
mysql> show tables;
Empty set (0.00 sec)
```

Dari tampilan baris terakhir diinformasikan tabel di dalam basis data **inventory** ialah **Empty set**, atau basis data **inventory** belum mempunyai tabel.

Untuk pembuatan suatu tabel, minimum perlu disiapkan satu atribut untuk tabel tersebut.

Misal diberikan data untuk tabel ruang adalah sebagai berikut:

ruang				
noruang	blokgedung	lantai	fungsi	luas
A101	A	1	Ruang Seminar	60
A102	A	1	Ruang Tamu	18
A201	A	2	Lab Komputer	72
B101	B	1	Ruang Kuliah	60
B303	B	3	Ruang Kuliah	72

Dengan menggunakan isi kolom atau field sebagaimana diberikan di atas, berikut adalah perintah-perintah untuk membuat tabel **ruang**.

Untuk mengecek hasil pembuatan tabel di atas, berikan perintah berikut:

--	--	--

```
mysql> show tables;
+-----+
| Tables_in_inventory |
+-----+
| ruang |
+-----+
1 row in set (0.00 sec)
```

Dari tampilan di atas terlihat tabel **ruang** sudah berhasil di-create oleh MySQL. Kemudian untuk melihat kolom-kolom di dalamnya, berikan perintah “**describe**” berikut:

```
mysql> describe ruang;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| noruang | char(4) | NO | PRI | NULL |       |
| blokgedung | char(1) | NO |     | NULL |       |
| lantai | int(10) unsigned | NO |     | NULL |       |
| fungsi | varchar(30) | YES |     | NULL |       |
| luas | int(10) unsigned | YES |     | NULL |       |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.00 sec)
```

Atau dapat juga digunakan perintah berikut:

```
mysql> show columns from ruang;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| noruang | char(4) | NO | PRI | NULL |       |
| blokgedung | char(1) | NO |     | NULL |       |
| lantai | int(10) unsigned | NO |     | NULL |       |
| fungsi | varchar(30) | YES |     | NULL |       |
| luas | int(10) unsigned | YES |     | NULL |       |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.00 sec)
```

Kedua perintah di atas memberikan hasil tampilan yang sama.

E. Meremajaan tabel

1. Menambah kolom baru

Misal tabel **ruang** yang semula mempunyai lima kolom akan ditambah dengan kolom **kapasitas** dengan type data didalamnya berupa numerik tidak bertanda.

Perintah yang diberikan ialah dengan “**alter table**” dan “**add**” sebagai berikut:

```
mysql> alter table ruang
-> add kapasitas int unsigned
->;
Query OK, 0 rows affected (1.13 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

Kemudian untuk melihat hasil penambahan kolom **kapasitas**, tampilkan kolom-kolom di dalam tabel **ruang** sebagai berikut.

```
mysql> describe ruang;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| noruang | char(4) | NO | PRI | NULL |       |
| blokgedung | char(1) | NO |     | NULL |       |
| lantai | int(10) unsigned | NO |     | NULL |       |
| fungsi | varchar(30) | YES |     | NULL |       |
| luas | int(10) unsigned | YES |     | NULL |       |
| kapasitas | int(10) unsigned | YES |     | NULL |       |
+-----+-----+-----+-----+-----+-----+
6 rows in set (0.00 sec)
```

Penambahan kolom-kolom lainnya lakukan dengan cara yang sama.

2. Mengganti nama kolom

Misal untuk mengganti kolom **luas** dari tabel **ruang** dengan nama baru yaitu **luaslantai**, gunakan perintah “**change**” kemudian hasilnya dicek dengan perintah “**describe**”.

```
mysql> alter table ruang
-> change luas luaslantai int unsigned
->;
Query OK, 0 rows affected (0.92 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> describe ruang;
+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+
| noruang | char(4) | NO | PRI | NULL |
| blokgedung | char(1) | NO | NULL |
| lantai | int(10) unsigned | NO | NULL |
| fungsi | varchar(30) | YES | NULL |
| luaslantai | int(10) unsigned | YES | NULL |
| kapasitas | int(10) unsigned | YES | NULL |
+-----+-----+-----+-----+-----+
6 rows in set (0.03 sec)
```

Dari tampilan di atas ternyata kolom **luas** sudah berganti nama dengan **luaslantai**. Penggantian nama suatu kolom juga dapat sekaligus dengan mengganti type data dari kolom tersebut.

3. Mengganti lebar dan jenis kolom

Jika penggantian hanya meliputi lebar dan jenis kolom, perintah yang digunakan ialah “**modify**”.

Apabila kolom **lantai** type data akan diubah dari int(10) menjadi char(1), maka perintah yang digunakan ialah sebagai berikut.

```
mysql> alter table ruang
-> modify lantai char(1) not null
->;
Query OK, 0 rows affected (0.31 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

Selanjutnya untuk melihat hasilnya, berikan perintah “**describe**” berikut ini.

```
mysql> describe ruang;
+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+
| noruang | char(4) | NO | PRI | NULL |
| blokgedung | char(1) | NO | NULL |
| lantai | char(1) | NO | NULL |
| fungsi | varchar(30) | YES | NULL |
| luas | int(10) unsigned | YES | NULL |
| kapasitas | int(10) unsigned | YES | NULL |
+-----+-----+-----+-----+-----+
6 rows in set (0.00 sec)
```

Bandingkan tampilan isi kolom dari tabel **ruang** yang terakhir, dan bandingkan dengan tampilan sebelumnya.

4. Mengganti nama tabel

Penggantian nama tabel dimungkinkan, dengan menggunakan perintah “**rename**”, sebagai berikut:

--	--	--

```
mysql> alter table ruang
-> rename ruangkelas
->;
Query OK, 0 rows affected (0.59 sec)

mysql> show tables;
+-----+
| Tables_in_inventory |
+-----+
| ruangkelas           |
+-----+
1 row in set (0.00 sec)
```

sel

Atau dengan perintah yang lain, yaitu dengan “**rename table**”. Misal nama tabel **ruangkelas** akan dikembalikan menjadi **ruang**, dengan perintah sebagai berikut:

```
mysql> rename table ruangkelas
-> to ruang
->;
Query OK, 0 rows affected (0.11 sec)

mysql> show tables;
+-----+
| Tables_in_inventory |
+-----+
| ruang               |
+-----+
1 row in set (0.00 sec)
```

Dengan perintah yang terakhir diberikan, nama tabel diubah kembali menjadi **ruang**.

5. Menghapus table

Perintah untuk menghapus suatu tabel dengan MySQL ialah dengan “**drop table**”, sebagai berikut:

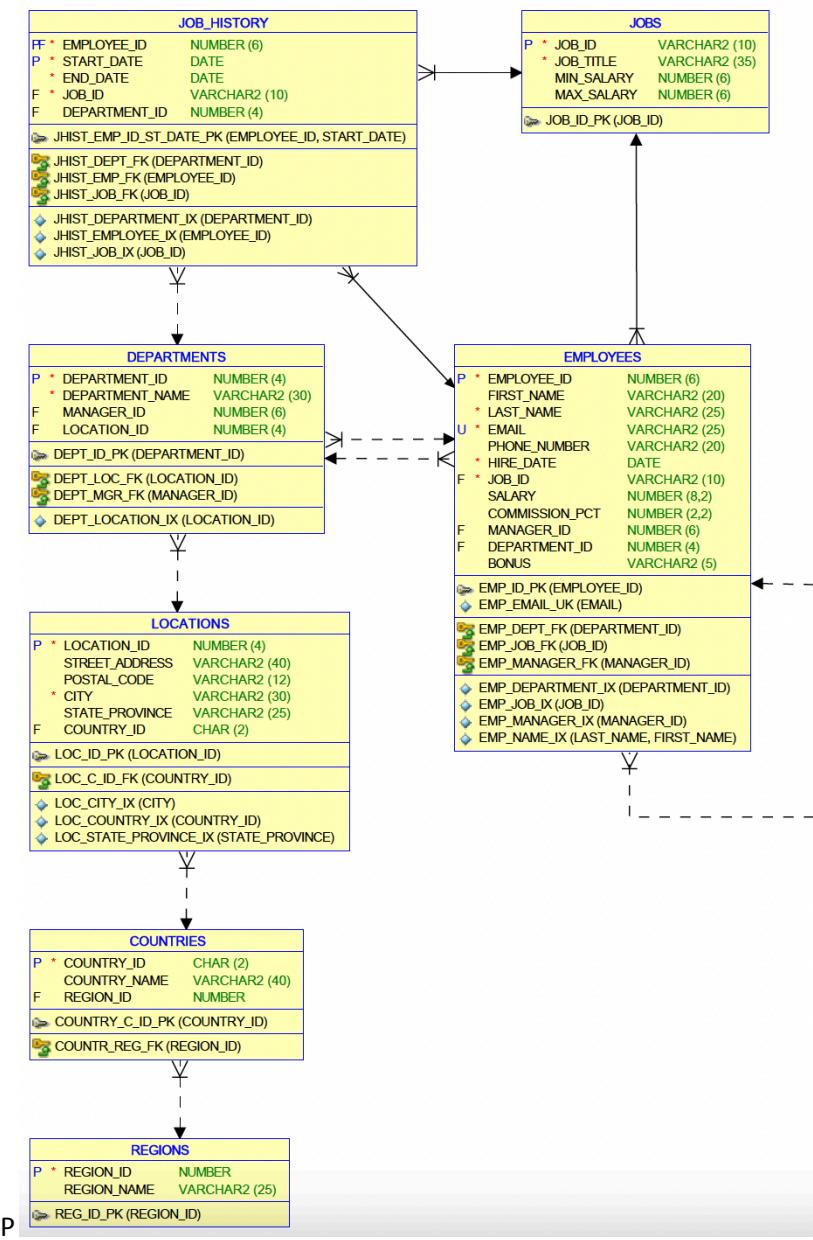
```
mysql> use inventory;
Database changed
mysql> show tables;
+-----+
| Tables_in_inventory |
+-----+
| ruang               |
+-----+
1 row in set (0.00 sec)

mysql> drop table ruang;
Query OK, 0 rows affected (0.16 sec)

mysql> show tables;
Empty set (0.00 sec)
```

Karena basis data **inventory** hanya memiliki satu tabel yaitu tabel **ruang**, maka hasil penghapusan tabel **ruang** akan menyebabkan **Empty set** untuk tabel di dalam basis data **inventory**.

Tugas:



Physical Data Model ini sama dengan phisical data model yang digunakan didalam perkuliahan. Untuk Tugas :

1. Buat DDL dan dieksekusi di lingkungan RDBMS MySQL.
2. Isi Data silakan diisi yang diambil dari data yg diberikan ditabel-tabel yang telah diberikan di kelas.

*catatan : DDL Oracle tidak bisa langsung digunakan sehingga mahasiswa dapat mengetahui perbedaannya.

EMPLO

Praktikum #6

Pemasukan dan Import Data

Standar Kompetensi	
Mahasiswa dapat memasukkan data ke dalam tabel dan melakukan import data dari sistem lain	

Kompetensi Dasar	Indikator
1. Mampu mengisikan data ke dalam tabel dan menampilkan hasilnya	a. Mengetahui cara menggunakan perintah INSERT b. Mampu mengisikan data dengan benar c. Mengetahui pemanfaatan perintah SELECT
2. Mampu menampilkan data berdasarkan kriteria tertentu	a. Mengetahui cara menampilkan data dengan berbagai predikat, menggunakan perintah SELECT dan WHERE b. Mengetahui cara mengurutkan data dengan perintah ORDER BY
3. Mampu melakukan import data dari sistem yang berbeda	a. Mampu mempersiapkan data dalam bentuk txt untuk diimport ke dalam MySQL. b. Mampu melakukan import data ke dalam MySQL.

A. Persiapan

Hasil pembuatan suatu tabel dengan perintah **create table** akan menghasilkan struktur tabel yang masih belum terisi data, dan siap diisi dengan data melalui proses pengisian data.

Pengisian data dapat dilakukan dengan menggunakan perintah yang ada di dalam MySQL, atau dengan melakukan import data yang telah dihasilkan dari sistem lain (misalnya data yang berupa text dan disimpan dengan ekstensi .txt).

Sebelum memasukkan data ke dalam tabel terlebih dahulu perlu diperhatikan hal-hal sebagai berikut:

- Jumlah kolom, dan urutan dari kolom.
- Type data setiap kolom.

Kedua hal tersebut di atas diperlukan untuk pedoman pada waktu pemasukan data, dan jika di dalam pemasukan data digunakan deskripsi yang berbeda dengan deskripsi dari tabel, akan menyebabkan pemasukan data menjadi gagal (muncul pesan adanya error).

Sebelum dilakukan pemasukan data ke suatu tabel, basis data dari tabel tersebut harus diaktifkan terlebih dahulu dengan menggunakan perintah **use**.

B. Pemasukan Data

Perintah MySQL yang digunakan untuk pemasukan data ke dalam tabel ialah perintah **insert into**. Terdapat sejumlah variasi penulisan yang dapat dipilih di dalam proses pemasukan data ke dalam tabel dengan menggunakan perintah tersebut.

Sebelumnya lihat kembali isi tabel **ruang** sebagaimana telah diberikan pada Praktikum #1. Untuk memasukkan data dari tabel tersebut dapat dipilih cara sebagai berikut.

Aktifkan basis data yang berisi tabel **ruang**, yaitu **inventory**.

```
mysql> use inventory;  
Database changed
```

Berikut diberikan tiga cara yang dapat dipilih di dalam memasukkan data ke dalam tabel **ruang**.

Cara-1:

Masukkan data record pertama dari tabel **ruang**.

```
mysql> insert into ruang  
-> set noruang="A101",  
-> blokgedung="A",  
-> lantai=1,  
-> fungsi="Ruang Seminar",  
-> luas=60  
->;  
Query OK, 1 row affected (0.50 sec)
```

Data yang mempunyai type non-numerik (char, varchar, date) ditulis dengan diapit oleh tanda “ atau ‘. Untuk mengecek hasil pemasukan data (seluruh kolom ditampilkan), gunakan perintah **select** sebagai berikut:

```
mysql> select * from ruang;  
+-----+-----+-----+-----+  
| noruang | blokgedung | lantai | fungsi | luas |  
+-----+-----+-----+-----+  
| A101 | A | 1 | Ruang Seminar | 60 |  
+-----+-----+-----+-----+  
1 row in set (0.00 sec)
```

Dari tampilan terakhir terlihat tabel ruang telah memiliki satu record atau baris.

Cara-2:

Setelah data baris pertama dari tabel **ruang** berhasil dimasukkan, selanjutnya data baris kedua akan dimasukkan dengan menggunakan cara sebagai berikut:

```
mysql> insert into ruang  
-> (noruang, blokgedung, lantai, fungsi, luas)  
-> values  
-> ("A102", "A", 1, "Ruang Tamu", 18)  
->;  
Query OK, 1 row affected (0.75 sec)
```

Kemudian untuk cek hasil pemasukan data yang ke dua, berikan perintah sebagai berikut:

```
mysql> select * from ruang;  
+-----+-----+-----+-----+  
| noruang | blokgedung | lantai | fungsi | luas |  
+-----+-----+-----+-----+  
| A101 | A | 1 | Ruang Seminar | 60 |  
| A102 | A | 1 | Ruang Tamu | 18 |  
+-----+-----+-----+-----+  
2 rows in set (0.00 sec)
```

Dari tampilan terakhir ditunjukkan adanya dua baris data di dalam tabel **ruang**.

Cara-3

Cara-3 menggunakan baris perintah yang lebih sederhana. Misal untuk baris ke tiga dari tabel **ruang** akan dimasukkan dengan cara yang ke tiga, perintahnya adalah sebagai berikut:

```
mysql> insert into ruang  
-> values ("A201", "A", 2, "Lab Komputer", 72)  
->;  
Query OK, 1 row affected (0.97 sec)
```

Kemudian untuk cek hasil pemasukan data yang ke tiga, berikan perintah sebagai berikut:

```

mysql> select * from ruang;
+-----+-----+-----+-----+-----+
| noruang | blokgedung | lantai | fungsi | luas |
+-----+-----+-----+-----+-----+
| A101    | A          | 1     | Ruang Seminar | 60  |
| A102    | A          | 1     | Ruang Tamu   | 18  |
| A201    | A          | 2     | Lab Komputer | 72  |
+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)

```

Dari tampilan terakhir ditunjukkan tabel **ruang** sudah berisi tiga baris data.

Pemasukan data dapat juga dilakukan untuk lebih dari satu baris data. Berikut adalah perintah untuk memasukkan dua baris data sekaligus.

```

mysql> insert into ruang
-> values ("B101","B",1,"Ruang Kuliah",60),
-> ("B303","B",3,"Ruang Kuliah",72)
->;
Query OK, 2 rows affected (0.58 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql> select * from ruang;
+-----+-----+-----+-----+-----+
| noruang | blokgedung | lantai | fungsi | luas |
+-----+-----+-----+-----+-----+
| A101    | A          | 1     | Ruang Seminar | 60  |
| A102    | A          | 1     | Ruang Tamu   | 18  |
| A201    | A          | 2     | Lab Komputer | 72  |
| B101    | B          | 1     | Ruang Kuliah | 60  |
| B303    | B          | 3     | Ruang Kuliah | 72  |
+-----+-----+-----+-----+-----+
5 rows in set (0.00 sec)

```

Hasil pemasukan data yang terakhir seperti biasa ditampilkan dengan perintah **select**. Untuk pemasukan data yang lebih banyak lagi gunakan cara yang sama.

C. Menampilkan Data dengan Kriteria Tertentu

Pada bagian sebelumnya telah dijelaskan penggunaan perintah **select *** untuk menampilkan isi semua kolom dari baris-baris yang ada pada suatu tabel.

Pengembangan berikutnya dari penggunaan perintah **select** ialah dengan menyertakan kriteria atau kondisi tertentu, yaitu:

- hanya untuk kolom-kolom tertentu,
- hanya untuk baris-baris yang memenuhi kriteria yang diberikan, atau
- kombinasi dari kedua kondisi di atas.

Untuk keperluan tersebut digunakan perintah **select** dan **where** sebagai berikut.

1. Menampilkan kolom-kolom tertentu

misal dari tabel **ruang** hanya akan ditampilkan tiga kolom, yaitu: noruang, fungsi, dan luas.

```

mysql> select noruang, fungsi, luas from ruang;
+-----+-----+-----+
| noruang | fungsi | luas |
+-----+-----+-----+
| A101    | Ruang Seminar | 60  |
| A102    | Ruang Tamu   | 18  |
| A201    | Lab Komputer | 72  |
| B101    | Ruang Kuliah | 60  |
| B303    | Ruang Kuliah | 72  |
+-----+-----+-----+
5 rows in set (0.00 sec)

```

Tampilan hanya berisi tiga kolom, dengan semua baris data ditampilkan.

2. Menampilkan baris yang memenuhi kriteria tertentu

Berikut hanya diminta ditampilkan data **ruang** yang mempunyai luas <= 50.

```

mysql> select * from ruang
-> where luas <= 50
-> ;
+-----+-----+-----+-----+-----+
| noruang | blokgedung | lantai | fungs i | luas |
+-----+-----+-----+-----+-----+
| A102   | A          | 1      | Ruang Tamu | 18   |
+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)

```

Hasil tampilannya hanya terdiri atas satu baris (yaitu yang memenuhi kriteria luas yang diberikan), dan tampilan menyajikan semua kolom (lima kolom) dari baris yang terpilih.

- Menampilkan kolom-kolom tertentu dari baris-baris yang memenuhi kriteria tertentu
Operasi untuk menampilkan kolom tertentu dan dengan baris yang memenuhi kriteria yang diberikan merupakan operasi kombinasi. Operasi akan diselesaikan dengan menggunakan perintah **select** yang dilengkapi dengan perintah **where**.

```

mysql> select noruang, fungs i, luas from ruang
-> where luas <= 50
-> ;
+-----+-----+-----+
| noruang | fungs i | luas |
+-----+-----+-----+
| A102   | Ruang Tamu | 18  |
+-----+-----+-----+
1 row in set (0.00 sec)

```

Bandingkan hasil tampilannya dengan dua tampilan sebelumnya.

- Menampilkan data yang diurutkan

Perintah yang digunakan untuk mengurutkan data ialah dengan perintah **order by**. Misal isi tabel **ruang** akan ditampilkan dan diurutkan menurut kolom **luas**, maka perintahnya adalah sebagai berikut:

```

mysql> select * from ruang
-> order by luas
-> ;
+-----+-----+-----+-----+-----+
| noruang | blokgedung | lantai | fungs i | luas |
+-----+-----+-----+-----+-----+
| A102   | A          | 1      | Ruang Tamu | 18   |
| A101   | A          | 1      | Ruang Seminar | 60   |
| B101   | B          | 1      | Ruang Kuliah | 60   |
| A201   | A          | 2      | Lab Komputer | 72   |
| B303   | B          | 3      | Ruang Kuliah | 72   |
+-----+-----+-----+-----+-----+
5 rows in set (0.00 sec)

```

Apabila diinginkan urutan bukan dari nilai terkecil ke terbesar, tapi dari nilai terbesar ke terkecil maka perintahnya ditambah dengan perintah **desc** artinya *descending*, sebagai berikut:

```

mysql> select * from ruang
-> order by luas desc
-> ;
+-----+-----+-----+-----+-----+
| noruang | blokgedung | lantai | fungs i | luas |
+-----+-----+-----+-----+-----+
| A201   | A          | 2      | Lab Komputer | 72   |
| B303   | B          | 3      | Ruang Kuliah | 72   |
| A101   | A          | 1      | Ruang Seminar | 60   |
| B101   | B          | 1      | Ruang Kuliah | 60   |
| A102   | A          | 1      | Ruang Tamu | 18   |
+-----+-----+-----+-----+-----+
5 rows in set (0.00 sec)

```

Bandingkan hasilnya dengan pengurutan sebelumnya.

D. Import Data dari Sistem Lain

Yang dimaksud dengan import data ialah mengambil atau memperoleh data yang dihasilkan dari sistem lain untuk digunakan di dalam MySQL, tanpa harus dilakukan pengetikan kembali. Sistem lain tersebut misalnya Notepad, MS Word, atau MS Excel.

Sebagai persiapan proses import data, dengan menggunakan **Notepad** atau **MS Word** (gunakan type fonts Courier New), ketik data yang akan diisikan ke dalam tabel **ruang** sebagai berikut.

```
"A101","A",1,"Ruang Seminar",60  
"A102","A",1,"Ruang Tamu",18  
"A201","A",2,"Lab Komputer",72  
"B101","B",1,"Ruang Kuliah",60  
"B303","B",3,"Ruang Kuliah",72
```

Simpan hasilnya di *folder* yang mudah diingat, beri nama dengan **data_ruang.txt** (misalnya di folder c:\data\). Jika diketik dengan menggunakan MS Word simpan dengan type Plain Text (pilih pada waktu save as).

Selanjutnya data tersebut (dalam format .txt), yang berisi sebanyak lima baris data akan dicoba diimport ke dalam tabel **ruang**.

Sebelumnya kosongkan isi tabel **ruang** dengan menghapus semua baris data yang ada di dalamnya, dengan perintah **delete from** sebagai berikut.

```
mysql> select * from ruang;  
+-----+-----+-----+-----+-----+  
| noruang | blokgedung | lantai | fungs i | luas |  
+-----+-----+-----+-----+-----+  
| A101 | A | 1 | Ruang Seminar | 60 |  
| A102 | A | 1 | Ruang Tamu | 18 |  
| A201 | A | 2 | Lab Komputer | 72 |  
| B101 | B | 1 | Ruang Kuliah | 60 |  
| B303 | B | 3 | Ruang Kuliah | 72 |  
+-----+-----+-----+-----+-----+  
5 rows in set (0.00 sec)  
  
mysql> delete from ruang;  
Query OK, 5 rows affected (0.39 sec)  
  
mysql> select * from ruang;  
Empty set (0.00 sec)
```

Bagian pertama dari kumpulan perintah di atas ialah untuk melakukan pengecekan isi tabel **ruang**, dan ternyata berisi lima baris data. Kemudian dengan perintah **delete from** semua data yang ada di dalam tabel **ruang** dihapus. Hasilnya diberikan pada bagian akhir dari tampilan di atas, yaitu tabel **ruang** dalam kondisi **Empty set** atau kosong (tidak memiliki data).

Setelah isi tabel ruang dikosongkan, selanjutnya lakukan import data dari file dalam format .txt yang telah dihasilkan dari tahapan persiapan, dengan menggunakan perintah sebagai berikut:

```
mysql> load data local infile 'c:\\data\\data_ruang.txt'  
-> into table ruang  
-> fields terminated by ','  
-> enclosed by '\"'  
-> lines terminated by '\\r\\n'  
-> ;
```

Kemudian cek hasilnya sebagaimana biasanya.

```

mysql> select * from ruang;
+-----+-----+-----+-----+-----+
| noruang | blokgedung | lantai | fungsi      | luas |
+-----+-----+-----+-----+-----+
| A101    | A          | 1     | Ruang Seminar | 60  |
| A102    | A          | 1     | Ruang Tamu   | 18  |
| A201    | A          | 2     | Lab Komputer | 72  |
| B101    | B          | 1     | Ruang Kuliah | 60  |
| B303    | B          | 3     | Ruang Kuliah | 72  |
+-----+-----+-----+-----+-----+
5 rows in set (0.00 sec)

```

Tugas :

1. Isikan data dari tabel-tabel yang telah dibuat pada praktikum pertama, dengan berbagai macam cara yang diajarkan diatas. Sebagaimana diberikan pada tugas Praktikum #1.
2. Berikan dan tunjukkan persamaan dan perbedaan antara RDBMS MySQL dan Oracle(via APEX) berkaitan dengan
 - a. Memasukkan data
 - b. Menampilkan beserta mengurutkan data.

`Praktikum #7

Manipulasi Tabel

Standar Kompetensi	
Mahasiswa dapat melakukan operasi untuk melakukan manipulasi terhadap tabel	

Kompetensi Dasar	Indikator
1. Mampu melakukan pemeliharaan data	a. Mampu menggunakan perintah UPDATE dan SET untuk melakukan pemeliharaan data dengan identitas tertentu b. Mampu menggunakan perintah UPDATE dan SET untuk melakukan pemeliharaan data secara serempak
2. Mampu melakukan pembandingan data	a. Mengetahui cara menggunakan operator pembanding (<, <=, >=, >) b. Mengetahui cara menggunakan operator logika: and, or, not c. Mampu menggunakan operator pembanding dan operator logika untuk manipulasi tabel
3. Mampu melakukan pengurutan data berdasarkan kriteria tertentu	a. Mengetahui cara menggunakan perintah order by untuk mengurutkan data b. Mampu menggunakan perintah order by untuk mengurutkan data dengan satu kriteria atau lebih

A. Pemeliharaan Data

Agar selalu *up-to-date* atau karena berkembangnya kebutuhan, seringkali data yang sudah disimpan di dalam tabel perlu diubah dan disesuaikan dengan kondisi yang baru. Hal tersebut dapat dilakukan dengan menggunakan perintah **update**. Perintah **update** dapat diberlakukan untuk baris tertentu atau untuk seluruh baris yang ada secara serempak.

Untuk suatu tabel dimungkinkan ada penambahan kolom, atau sebaliknya ada kolom yang dihapus (melalui suatu prosedur tertentu). Demikian pula dapat terjadi adanya penambahan baris atau penghapusan baris data.

Data suatu kolom dapat pula dilakukan pemeliharaan, misalnya: barang tertentu yang menempati suatu ruang jumlahnya berubah.

1. Menambah kolom

Misal tabel **ruang** yang berisi lima kolom, akan ditambah dengan kolom yang ke enam yaitu kolom **kapasitas** yang berisi data tentang kapasitas setiap ruang.

Kapasitas setiap ruang adalah sebagai berikut:

A101 : 40	B101 : 40
A102 : 6	B303 : 50
A201 : 50	

Perintah untuk menambah kolom baru (kolom **kapasitas**) ialah dengan **alter** dan **add** sebagai berikut.

```
mysql> alter table ruang
      -> add kapasitas int not null default 0
      ->;
Query OK, 5 rows affected (0.67 sec)
Records: 5  Duplicates: 0  Warnings: 0
```

Hasil penambahan kolom dicek dengan perintah sebagai berikut.

```
mysql> select * from ruang;
+-----+-----+-----+-----+-----+-----+
| noruang | blokgedung | lantai | fungs i | luas | kapasitas |
+-----+-----+-----+-----+-----+-----+
| A101    | A          | 1      | Ruang Seminar | 60   | 0        |
| A102    | A          | 1      | Ruang Tamu   | 18   | 0        |
| A201    | A          | 2      | Lab Komputer  | 72   | 0        |
| B101    | B          | 1      | Ruang Kuliah  | 60   | 0        |
| B303    | B          | 3      | Ruang Kuliah  | 72   | 0        |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.00 sec)
```

Kolom **kapasitas** sudah masuk ke tabel ruang, akan tetapi dengan isi data semua 0. Kenapa?

2. Mengganti data di dalam kolom

Untuk mengganti data digunakan perintah **update** dan **set** serta **where** untuk memberikan predikat dari baris data yang akan diproses (diganti datanya).

Ambil kembali tabel **ruang** yang telah ditambah kolomnya dengan kolom **kapasitas**. Selanjutnya untuk menyesuaikan isi tabel dengan data **kapasitas** yang benar, berikut akan dilakukan penggantian data **kapasitas** dari ruang A101 dan A102 dengan perintah **update**.

```
mysql> update ruang
      -> set kapasitas=40
      -> where noruang="A101"
      -> ;
Query OK, 1 row affected (0.41 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> update ruang
      -> set kapasitas=6
      -> where noruang="A102"
      -> ;
Query OK, 1 row affected (0.50 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

Hasilnya disajikan pada tampilan berikut.

```
mysql> select * from ruang;
+-----+-----+-----+-----+-----+-----+
| noruang | blokgedung | lantai | fungs i | luas | kapasitas |
+-----+-----+-----+-----+-----+-----+
| A101    | A          | 1      | Ruang Seminar | 60   | 40       |
| A102    | A          | 1      | Ruang Tamu   | 18   | 6        |
| A201    | A          | 2      | Lab Komputer  | 72   | 0        |
| B101    | B          | 1      | Ruang Kuliah  | 60   | 0        |
| B303    | B          | 3      | Ruang Kuliah  | 72   | 0        |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.00 sec)
```

Dengan cara yang sama lakukan untuk baris-baris lain yang akan diganti datanya.

Penggantian data secara serempak (untuk seluruh baris) pada suatu tabel dilakukan dengan perintah **update**, tetapi tanpa menambahkan perintah **where**.

Jadi perintah **where** akan membatasi akibat yang ditimbulkan dari **update**, hanya pada baris-baris yang memenuhi identitas atau predikat yang diberikan.

B. Pembandingan Data

Untuk beberapa keperluan sering kali suatu data akan dibandingkan dengan data yang lain (berupa isi suatu kolom atau suatu konstanta).

Contoh dari proses pembandingan tersebut ialah:

- Menampilkan ruang-ruang di lantai 3 yang digunakan untuk ruang kuliah dengan kapasitas yang dapat menampung 40 peserta.

- Menampilkan ruang seminar (tanpa membedakan dari lantai mana), yang mempunyai luas minimum 40.

Pembandingan akan lebih komplek lagi dengan menggabungkan lebih dari satu data yang dipakai sebagai dasar pembandingan, sebagaimana dicontohkan berikut ini:

- Menampilkan nomor dan nama barang yang diperoleh dari sumber dana PNBP dan diproduksi pada tahun 2008.
- Menampilkan nomor dan nama barang yang diperoleh dari sumber dana PNBP atau BMOM dan diproduksi pada tahun 2009.

Untuk keperluan tersebut digunakan operator pembanding, dan operator logika sebagai berikut:

Operator Pembanding

Operator	Arti
>	Lebih besar
<	Lebih kecil
=	Sama dengan
>=	Lebih besar atau sama dengan
<=	Lebih kecil atau sama dengan
<>	Tidak sama dengan

Operator Logika

Operator	Arti
AND atau &&	Dan
OR atau	Atau
NOT atau !	Tidak

Untuk penggunaan operator pembandingan di atas, lakukan tahapan-tahapan sebagai berikut:

1. Persiapan

Tambahkan data berikut ke dalam tabel **ruang** dan tabel **barang**.

ruang

noruang	blokgedung	lantai	fungsi	luas	kapasitas
B102	B	1	Ruang Kuliah	60	45
B201	B	2	Ruang Kuliah	75	65
B202	B	2	Ruang Kuliah	75	65
B203	B	2	Lab Komputer	60	50

barang

nobaran g	namabarang	sumberdana	thnproduksi	satuan	
208002	Kursi Kuliah	BMOM	2008	Unit	
210002	Kursi Kuliah	Hibah	2010	Unit	
211002	Kursi Kuliah	PNBP	2011	Unit	
509001	Komputer Desktop	PNBP	2009	Unit	
509002	Printer Laser	PNBP	2009	Unit	

Cek hasilnya dengan perintah **select**.

Untuk tabel **ruang** yang sudah di-update isinya adalah sebagai berikut:

```
mysql> select * from ruang;
+-----+-----+-----+-----+-----+-----+
| noruang | blokgedung | lantai | fungsi | luas | kapasitas |
+-----+-----+-----+-----+-----+-----+
| A101    | A          | 1     | Ruang Seminar | 60   | 40      |
| A102    | A          | 1     | Ruang Tamu   | 18   | 6       |
| A201    | A          | 2     | Lab Komputer | 72   | 50      |
| B101    | B          | 1     | Ruang Kuliah | 60   | 40      |
| B102    | B          | 1     | Ruang Kuliah | 60   | 45      |
| B201    | B          | 2     | Ruang Kuliah | 70   | 50      |
| B202    | B          | 2     | Ruang Kuliah | 75   | 60      |
| B203    | B          | 2     | Lab Komputer | 50   | 40      |
| B303    | B          | 3     | Ruang Kuliah | 72   | 50      |
+-----+-----+-----+-----+-----+-----+
9 rows in set (0.00 sec)
```

Dan isi tabel **barang** yang sudah di-update adalah sebagaimana tampilan berikut ini:

```
mysql> select * from barang;
+-----+-----+-----+-----+-----+
| nobarang | namabarang | sumberdana | thnproduksi | satuan |
+-----+-----+-----+-----+-----+
| 108001  | Meja Kuliah | PNBP        | 2008         | Unit    |
| 109001  | Meja Komputer | PNBP        | 2009         | Unit    |
| 110001  | Meja Sidang   | PNBP        | 2010         | Unit    |
| 111001  | Meja Tamu    | BM0M        | 2011         | Unit    |
| 208001  | Kursi Kuliah | PNBP        | 2008         | Unit    |
| 208002  | Kursi Kuliah | BM0M        | 2008         | Unit    |
| 209001  | Kursi Lipat   | PNBP        | 2009         | Unit    |
| 210001  | Kursi Sidang  | PNBP        | 2010         | Unit    |
| 210002  | Kursi Kuliah | Hibah       | 2010         | Unit    |
| 211001  | Kursi Tamu   | BM0M        | 2011         | Unit    |
| 211002  | Kursi Kuliah | PNBP        | 2011         | Unit    |
| 308001  | LCD           | PNBP        | 2008         | Unit    |
| 309001  | LCD           | Hibah       | 2009         | Unit    |
| 409001  | Layar Peraga  | BM0M        | 2009         | Unit    |
| 409002  | Layar Peraga  | Hibah       | 2009         | Unit    |
| 409003  | Whiteboard    | PNBP        | 2009         | Unit    |
| 509001  | Komputer Desktop | PNBP        | 2009         | Unit    |
| 509002  | Printer Laser | PNBP        | 2009         | Unit    |
+-----+-----+-----+-----+-----+
18 rows in set (0.00 sec)
```

2. Penggunaan perintah pembandingan dan logika

- Menampilkan data dari tabel **ruang** untuk ruang-ruang yang berfungsi sebagai ruang kuliah, dan mempunyai kapasitas ≥ 50

```
mysql> select * from ruang
-> where fungsi="Ruang Kuliah" and kapasitas >= 50
-> ;
+-----+-----+-----+-----+-----+
| noruang | blokgedung | lantai | fungsi | luas | kapasitas |
+-----+-----+-----+-----+-----+
| B201    | B          | 2     | Ruang Kuliah | 70   | 50      |
| B202    | B          | 2     | Ruang Kuliah | 75   | 60      |
| B303    | B          | 3     | Ruang Kuliah | 72   | 50      |
+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)
```

Karena menggunakan perintah **select *** maka semua kolom yang ada pada tabel **ruang** akan ditampilkan (untuk baris yang memenuhi kriteria).

- Menampilkan data **noruang**, **blokgedung**, dan **fungsi** dari tabel **ruang** untuk ruang-ruang yang berlokasi di lantai 1 dan berfungsi sebagai ruang kuliah.

```
mysql> select noruang, blokgedung, fungsi from ruang
-> where lantai=1 and fungsi="Ruang Kuliah"
-> ;
+-----+-----+-----+
| noruang | blokgedung | fungsi |
+-----+-----+-----+
| B101    | B          | Ruang Kuliah |
| B102    | B          | Ruang Kuliah |
+-----+-----+-----+
2 rows in set (0.00 sec)
```

Berbeda dengan contoh sebelumnya, untuk contoh ini hanya kolom tertentu yang ditampilkan sesuai dengan yang disebut pada perintah **select**.

- c. Menampilkan data dari tabel **barang** untuk yang mempunyai tahun produksi ≤ 2009 dan menggunakan sumber dana **BMOM** atau **Hibah**.

```
mysql> select * from barang
-> where thnproduksi <= 2009 and (sumberdana="BMOM" or sumberdana="Hibah")
-> ;
+-----+-----+-----+-----+-----+
| nobarang | namabarang | sumberdana | thnproduksi | satuan |
+-----+-----+-----+-----+-----+
| 208002 | Kursi Kuliah | BMOM | 2008 | Unit |
| 309001 | LCD | Hibah | 2009 | Unit |
| 409001 | Layar Peraga | BMOM | 2009 | Unit |
| 409002 | Layar Peraga | Hibah | 2009 | Unit |
+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)
```

Atau dengan logika lain, untuk memenuhi permintaan yang terakhir dapat digunakan perintah sebagai berikut:

```
mysql> select * from barang
-> where thnproduksi <= 2009 and sumberdana <> "PNBP"
-> ;
+-----+-----+-----+-----+-----+
| nobarang | namabarang | sumberdana | thnproduksi | satuan |
+-----+-----+-----+-----+-----+
| 208002 | Kursi Kuliah | BMOM | 2008 | Unit |
| 309001 | LCD | Hibah | 2009 | Unit |
| 409001 | Layar Peraga | BMOM | 2009 | Unit |
| 409002 | Layar Peraga | Hibah | 2009 | Unit |
+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)
```

Bandingkan hasilnya, ternyata menampilkan hasil yang sama. Perintah dapat diberikan lebih singkat, kenapa?

C. Pengurutan Data

Pada bagian sebelumnya diberikan cara bagaimana menampilkan isi suatu tabel sebagai hasil manipulasi, yaitu:

- Semua kolom, untuk baris-baris yang memenuhi satu atau lebih kriteria.
- Kolom tertentu, untuk baris-baris yang memenuhi satu atau lebih kriteria.

Selanjutnya untuk mempermudah pembacaan hasil yang ditampilkan, dapat dilakukan pengurutan hasil tampilan berdasarkan satu atau lebih kriteria pengurutan.

1. Pengurutan dengan satu kriteria

Sebagai contoh misal akan ditampilkan data dari tabel barang yang bukan **Kursi Kuliah** dengan susunan urut menurut **sumberdana** (berdasarkan urutan abjad).

```
mysql> select * from barang
-> where namabarang <> "Kursi Kuliah"
-> order by sumberdana
-> ;
+-----+-----+-----+-----+-----+
| nobarang | namabarang | sumberdana | thnproduksi | satuan |
+-----+-----+-----+-----+-----+
| 111001 | Meja Tamu | BMOM | 2011 | Unit |
| 409001 | Layar Peraga | BMOM | 2009 | Unit |
| 211001 | Kursi Tamu | BMOM | 2011 | Unit |
| 409002 | Layar Peraga | Hibah | 2009 | Unit |
| 309001 | LCD | Hibah | 2009 | Unit |
| 108001 | Meja Kuliah | PNBP | 2008 | Unit |
| 509001 | Komputer Desktop | PNBP | 2009 | Unit |
| 409003 | Whiteboard | PNBP | 2009 | Unit |
| 308001 | LCD | PNBP | 2008 | Unit |
| 210001 | Kursi Sidang | PNBP | 2010 | Unit |
| 209001 | Kursi Lipat | PNBP | 2009 | Unit |
| 110001 | Meja Sidang | PNBP | 2010 | Unit |
| 109001 | Meja Komputer | PNBP | 2009 | Unit |
| 509002 | Printer Laser | PNBP | 2009 | Unit |
+-----+-----+-----+-----+-----+
14 rows in set (0.00 sec)
```

Amati hasilnya, ternyata meskipun sudah urut menurut **sumberdana** akan tetapi **thnproduksi** masih acak (belum urut).

2. Pengurutan dengan dua kriteria

Jika diinginkan tampilan diurutkan berdasarkan lebih dari satu dasar pengurutan, maka kolom-kolom yang digunakan sebagai dasar pengurutan dituliskan berurutan dengan diberi batas berupa tanda koma.

Jika tampilan terakhir dari tabel **barang** akan diurutkan berdasarkan **sumberdana** dan **thnproduksi**, maka perintahnya adalah sebagai berikut.

```
mysql> select * from barang
-> where namabarang <> "Kursi Kuliah"
-> order by sumberdana, thnproduksi
-> ;
+-----+-----+-----+-----+-----+
| nobarang | namabarang | sumberdana | thnproduksi | satuan |
+-----+-----+-----+-----+-----+
| 409001 | Layar Peraga | BMOM | 2009 | Unit |
| 111001 | Meja Tamu | BMOM | 2011 | Unit |
| 211001 | Kursi Tamu | BMOM | 2011 | Unit |
| 409002 | Layar Peraga | Hibah | 2009 | Unit |
| 309001 | LCD | Hibah | 2009 | Unit |
| 108001 | Meja Kuliah | PNBP | 2008 | Unit |
| 308001 | LCD | PNBP | 2008 | Unit |
| 209001 | Kursi Lipat | PNBP | 2009 | Unit |
| 109001 | Meja Komputer | PNBP | 2009 | Unit |
| 409003 | Whiteboard | PNBP | 2009 | Unit |
| 509001 | Komputer Desktop | PNBP | 2009 | Unit |
| 509002 | Printer Laser | PNBP | 2009 | Unit |
| 210001 | Kursi Sidang | PNBP | 2010 | Unit |
| 110001 | Meja Sidang | PNBP | 2010 | Unit |
+-----+-----+-----+-----+-----+
14 rows in set (0.00 sec)
```

Bandingkan hasilnya, apakah sudah sesuai dengan yang diinginkan?

Jika diinginkan tampilan akan disusun dengan lebih dari dua dasar pengurutan, lakukan perluasan dari perintah **order by** yang diberikan terakhir.

Tugas

Dengan menggunakan tabel **ruang** dan tabel **barang** yang terakhir (setelah di-update), susun perintah untuk menghasilkan tampilan-tampilan sebagai berikut.

1. Menampilkan data dari tabel **ruang** untuk ruang-ruang selain “**Ruang Seminar**” dan selain “**Ruang Tamu**” dengan luas tidak kurang dari **50**.
2. Menampilkan nomor ruang, fungsi, luas, dan kapasitas untuk “**Ruang Kuliah**” yang mempunyai **luas** paling sedikit 1,5 dari **kapasitas**.
3. Menampilkan data dari tabel **barang** yang menggunakan sumber dana “**Hibah**” atau “**BMOM**” akan tetapi yang bukan berupa “**Kursi Kuliah**”.
4. Menampilkan nomor barang, nama barang, sumber dana, dan tahun produksi dengan sumber dana “**PNBP**” dan dengan tahun produksi bukan “**2009**”.
5. Menampilkan data dari tabel **barang** untuk semua barang dengan tahun produksi “**2008**”, dan bukan berupa “**Kursi Kuliah**” dari sumber dana “**PNBP**”.
6. Menampilkan data dari tabel **ruang** diurutkan dari **luas** yang terbesar ke yang terkecil, dan untuk ruang dengan **luas** sama besar, tampilkan terlebih dahulu yang mempunyai **kapasitas** lebih besar.
7. Menampilkan data dari tabel **barang** diurutkan menurut tahun produksi yang paling baru ke yang sebelumnya, dan jika tahun produksinya sama tampilkan terlebih dahulu menurut urutan abjad dari sumber dana.

-oOo-

--	--	--

Praktikum #8

Pengolahan Data Numerik dan Non-Numerik

Standar Kompetensi

Mahasiswa dapat melakukan operasi pengolahan data numerik (pengolahan statistik dasar) dan data non-numerik (pengolahan string).

Kompetensi Dasar	Indikator
1. Mampu menggunakan fungsi statistik dasar untuk pengolahan data numerik	a. Mampu menggunakan fungsi count untuk menghitung jumlah data. b. Mampu menggunakan fungsi avg untuk mendapatkan nilai rata-rata. c. Mampu menggunakan fungsi min untuk mendapatkan angka minimum. d. Mampu menggunakan fungsi max untuk mendapatkan angka maksimum. e. Mampu menggunakan fungsi sum untuk menjumlahkan angka yang ada di dalam field tertentu.
2. Mampu melaksanakan pengolahan data non-numerik atau string	a. Mampu menggunakan operator like untuk memperoleh data dengan kriteria tertentu berupa sub-string dari isi suatu field. b. Mampu menggunakan operator regexp untuk operasi pengolahan yang lebih specific, dengan menggunakan kriteria berupa huruf, dan sub-string yang terletak pada berbagai posisi di dalam string (depan, belakang, atau sembarang) c. Mampu menggunakan operator regexp untuk menampilkan data dengan kriteria yang digunakan berupa panjang string.

A. Pengolahan Data

Data di dalam tabel dapat dimanfaatkan lebih lanjut untuk diolah menjadi informasi melalui tahapan pengolahan.

Diantara pengolahan yang seringkali dilakukan untuk data numerik ialah pengolahan untuk mendapatkan angka-angka statistik, dengan menggunakan fungsi-fungsi sebagai berikut:

- Fungsi count : menghitung jumlah data
- Fungsi max : mencari nilai maksimum
- Fungsi min : mencari nilai minimum
- Fungsi avg : menghitung nilai rata-rata
- Fungsi sum : menghitung jumlah sekumpulan angka dari suatu field

Sedangkan untuk data non-numerik berupa string, disediakan juga sejumlah operator untuk mendukung pengolahan yang dilakukan.

Operator yang dapat digunakan untuk keperluan tersebut ialah sebagai berikut:

- Operator like : digunakan untuk memperoleh data yang “menyerupai” atau “hampir sama” dengan kriteria yang diberikan
- Operator regexp : berfungsi hampir sama dengan operator like, dan penggunaannya

--	--	--

ditambah dengan simbol-simbol berikut:

- Simbol titik (.) untuk mewakili satu karakter
- Simbol [?] untuk mewakili beberapa karakter atau range yang ditentukan
- Simbol ^ untuk posisi awal dari kriteria
- Simbol \$ untuk menandai posisi akhir dari kriteria

B. Pengolahan statistik dasar data numerik.

Pengolahan statistik dasar merupakan pengolahan yang akan menghasilkan suatu nilai tunggal sebagai hasil pengolahan terhadap sekumpulan data numerik.

Untuk mencoba penggunaan fungsi-fungsi untuk pengolahan statistika dasar, ambil kembali tabel-tabel yang sudah dihasilkan, yaitu tabel-tabel **ruang** dan **barang**.

1. Menampilkan berapa jumlah ruang yang berfungsi sebagai “Lab Komputer” dari tabel **ruang**.

```
mysql> select count(*) from ruang
      -> where fungs_i = "Lab Komputer";
+-----+
| count(*) |
+-----+
|      2   |
+-----+
1 row in set (0.00 sec)
```

2. Menampilkan kapasitas maksimal dari “Ruang Kuliah” yang berlokasi pada lantai 2, yang datanya disimpan di dalam tabel **ruang**.

```
mysql> select max(kapasitas) from ruang
      -> where fungs_i = "Ruang Kuliah";
+-----+
| max(kapasitas) |
+-----+
|        60      |
+-----+
1 row in set (0.00 sec)
```

3. Menampilkan tahun produksi paling tua dari barang yang disimpan di dalam tabel **barang**, dengan sumber dana dari “PNBP”.

```
mysql> select min(thnproduksi) from barang
      -> where sumberdana = "PNBP";
+-----+
| min(thnproduksi) |
+-----+
|        2008      |
+-----+
1 row in set (0.00 sec)
```

4. Menampilkan rata-rata luas ruang yang berfungsi sebagai “Ruang Kuliah” yang terletak di lantai 2, yang datanya disimpan di dalam tabel **ruang**.

--	--	--

```
mysql> select avg(luas) from ruang
-> where fungsi = "Ruang Kuliah"
-> and lantai = 2;
+-----+
| avg(luas) |
+-----+
| 72.5000 |
+-----+
1 row in set (0.00 sec)
```

C. Pengolahan untuk data string

Kalau untuk data numerik pengolahan akan terkait dengan operasi aritmatika (meskipun dapat juga operasi pengurutan), sedangkan untuk data string pengolahan lebih terkait pada pencarian data dengan berbagai kriteria.

Di dalam praktikum ini kriteria yang diberikan merupakan perluasan dari contoh-contoh kriteria yang telah diberikan sebelumnya.

Misal untuk ruang terdapat berbagai fungsi lab, yaitu:

- Lab Komputer
- Lab Statistik
- Lab Matematika
- Lab Fisika

maka untuk menampilkan ruang-ruang di atas cukup dengan menyebut ruang dengan tiga karakter pertama dari fungsi ruang ialah “Lab”. Atau bisa juga potongan kata yang digunakan sebagai criteria terletak di tengah, di akhir, atau pada posisi sembarang dari data.

Untuk keperluan tersebut digunakan operator regexp (REGuler EXPression) dikombinasikan dengan simbol-simbol sebagaimana telah dijelaskan pada bagian sebelumnya.

Berikut adalah contoh-contoh untuk operasi tersebut.

1. Menampilkan data dari tabel **barang** untuk semua barang berupa meja, yang berarti akan dikenali dari bagian depan nama barang berupa tulisan “Meja”.

```
mysql> select * from barang
-> where namabarang like "Meja%";
```

nobarang	namabarang	sumberdana	thnproduksi	satuan
108001	Meja Kuliah	PNBP	2008	Unit
109001	Meja Komputer	PNBP	2009	Unit
110001	Meja Sidang	PNBP	2010	Unit
111001	Meja Tamu	BMOM	2011	Unit

4 rows in set (0.00 sec)

2. Menampilkan nomor barang, nama barang, dan sumber dana dari tabel **barang** untuk semua barang yang digunakan untuk kuliah (ditandai dengan terdapatnya string “Kuliah” di dalam penulisan nama barang).

```
mysql> select nobarang, namabarang, sumberdana from barang
-> where namabarang like "%Kuliah%";
```

nobarang	namabarang	sumberdana
108001	Meja Kuliah	PNBP
208001	Kursi Kuliah	PNBP
208002	Kursi Kuliah	BMOM
210002	Kursi Kuliah	Hibah
211002	Kursi Kuliah	PNBP

5 rows in set (0.00 sec)

Atau karena tulisan “Kuliah” ternyata semua terletak pada bagian akhir dari nama barang, maka penulisan operator **like** berikut juga memberikan hasil yang sama.

```
mysql> select nobarang, namabarang, sumberdana from barang
-> where namabarang like "%kuliah";
+-----+-----+-----+
| nobarang | namabarang | sumberdana |
+-----+-----+-----+
| 108001  | Meja Kuliah | PNBP
| 208001  | Kursi Kuliah | PNBP
| 208002  | Kursi Kuliah | BMOM
| 210002  | Kursi Kuliah | Hibah
| 211002  | Kursi Kuliah | PNBP
+-----+-----+-----+
5 rows in set (0.00 sec)
```

Coba kembali penggunaan operator like yang terakhir yaitu: **like “%kuliah”** dengan ditulis sebagai **like “%Kuliah”**. Bagaimana hasilnya? Ternyata sama.

Untuk memperketat hasil pencarian agar sesuai persis dengan kriteria yang diberikan, gunakan perintah **like binary** sebagaimana contoh berikut.

```
mysql> select nobarang, namabarang, sumberdana from barang
-> where namabarang like binary "%kuliah";
Empty set (0.41 sec)
```

Keapa hasilnya **Empty Set**? Jawabannya adalah karena tidak ada tulisan “**kuliah**” (semua ditulis dengan huruf kecil), pada nama barang. Yang ada adalah tulisan “**Kuliah**”, yaitu dengan huruf pertama ditulis dengan huruf besar.

Coba lagi dengan **like binary “%Kuliah”**, bandingkan hasilnya.

3. Menampilkan nomor barang, nama barang, dan sumber dana dari tabel **barang** untuk barang berupa meja, kursi dan komputer.

Melihat variasi nama barang di dalam data tabel **barang** yang sekarang , maka untuk memenuhi permintaan tersebut dapat dilakukan dengan memberikan kategori berupa barang yang mempunyai nama depan berupa huruf “M” atau “K”.

Dengan menggunakan perintah regexp, perintahnya adalah sebagai berikut.

```
mysql> select nobarang, namabarang, sumberdana from barang
-> where namabarang regexp "^[K"
-> or namabarang regexp "^[M";
+-----+-----+-----+
| nobarang | namabarang | sumberdana |
+-----+-----+-----+
| 108001  | Meja Kuliah | PNBP
| 109001  | Meja Komputer | PNBP
| 110001  | Meja Sidang | PNBP
| 111001  | Meja Tamu | BMOM
| 208001  | Kursi Kuliah | PNBP
| 208002  | Kursi Kuliah | BMOM
| 209001  | Kursi Lipat | PNBP
| 210001  | Kursi Sidang | PNBP
| 210002  | Kursi Kuliah | Hibah
| 211001  | Kursi Tamu | BMOM
| 211002  | Kursi Kuliah | PNBP
| 509001  | Komputer Desktop | PNBP
+-----+-----+-----+
12 rows in set (0.41 sec)
```

Bedakan tampilan yang dihasilkan jika diberikan perintah berikut.

--	--	--

```
mysql> select nobarang, namabarang, sumberdana from barang
-> where namabarang regexp "^[K-M]"
-> order by namabarang;
+-----+-----+-----+
| nobarang | namabarang | sumberdana |
+-----+-----+-----+
| 509001  | Komputer Desktop | PNBP
| 211002  | Kursi Kuliah | PNBP
| 210002  | Kursi Kuliah | Hibah
| 208002  | Kursi Kuliah | BMOM
| 208001  | Kursi Kuliah | PNBP
| 209001  | Kursi Lipat | PNBP
| 210001  | Kursi Sidang | PNBP
| 211001  | Kursi Tamu | BMOM
| 409002  | Layar Peraga | Hibah
| 409001  | Layar Peraga | BMOM
| 308001  | LCD | PNBP
| 309001  | LCD | Hibah
| 109001  | Meja Komputer | PNBP
| 108001  | Meja Kuliah | PNBP
| 110001  | Meja Sidang | PNBP
| 111001  | Meja Tamu | BMOM
+-----+-----+-----+
16 rows in set (0.00 sec)
```

Kenapa LCD ikut ditampilkan? Jawabannya ialah karena dengan **namabarang regexp “^[K-M]”** maka semua data dengan nama barang diawali dengan huruf K sampai dengan M (termasuk L) akan terpilih untuk ditampilkan.

4. Menampilkan nomor barang, nama barang, dan sumber dana dari tabel **barang** untuk barang dengan nama barang yang diakhiri dengan tulisan **“Kuliah”**. Tampilan diurutkan menurut nama barang.

```
mysql> select nobarang, namabarang, sumberdana from barang
-> where namabarang regexp "Kuliah$"
-> order by namabarang;
+-----+-----+-----+
| nobarang | namabarang | sumberdana |
+-----+-----+-----+
| 208001  | Kursi Kuliah | PNBP
| 208002  | Kursi Kuliah | BMOM
| 210002  | Kursi Kuliah | Hibah
| 211002  | Kursi Kuliah | PNBP
| 108001  | Meja Kuliah | PNBP
+-----+-----+-----+
5 rows in set (0.00 sec)
```

5. Menampilkan nomor barang, nama barang, dan sumber dana dari tabel **barang** untuk barang dengan penulisan nama barang yang mempunyai panjang 13 karakter.

```
mysql> select nobarang, namabarang, sumberdana from barang
-> where namabarang regexp "^.{13}$";
+-----+-----+-----+
| nobarang | namabarang | sumberdana |
+-----+-----+-----+
| 109001  | Meja Komputer | PNBP
| 509002  | Printer Laser | PNBP
+-----+-----+-----+
2 rows in set (0.00 sec)
```

Perhatikan jumlah titik yang diapit oleh tanda ^ dan \$ menunjukkan jumlah karakter yang diinginkan, yaitu tiga belas.

Perintah di atas dapat juga dituliskan dengan susunan sebagai berikut.

```
mysql> select nobarang, namabarang, sumberdana from barang
-> where namabarang regexp "^.{13}$";
+-----+-----+-----+
| nobarang | namabarang | sumberdana |
+-----+-----+-----+
| 109001  | Meja Komputer | PNBP
| 509002  | Printer Laser | PNBP
+-----+-----+-----+
2 rows in set (0.00 sec)
```

--	--	--

Perintah yang diberikan terakhir lebih praktis dan lebih mudah di dalam penulisannya dibandingkan dengan penulisan perintah sebelumnya.

Tugas 1

Perhatikan tabel **isiruang** yang telah Saudara hasilkan di Praktikum #2. Field apa yang digunakan sebagai kunci primernya?

Kolom pertama dari **nobarang** adalah sebagai berikut:

Angka 1: meja

Angka 3: LCD

Angka 2: kursi

Angka 4: layar peraga/whiteboard

Sedangkan kolom ke 2 dan 3 dari **nobarang** adalah tahun produksi.

Susun perintah untuk pengolahan-pengolahan sebagai berikut:

1. Menampilkan **noruang** dan **nobarang** dari tabel **isiruang**, yang di dalam ruang terkait terdapat Layar Peraga atau Whiteboard.
2. Menghitung jumlah LCD yang dimiliki, oleh ruang-ruang yang ada di dalam tabel **isiruang**.
3. Menampilkan jumlah kursi terbanyak pada ruang yang berlokasi di **blokgedung A**, yang datanya disimpan di dalam tabel **isiruang**.

TUGAS 2

Gunakan Database Organisasi, untuk membuat query dari permasalahan berikut

1. Hitunglah jumlah karyawan dan total gajinya yang memiliki ID departemen 90
2. Hitunglah rata-rata gaji karyawan yang memiliki ID pegawai antara 120 dan 200
3. Carilah gaji karyawan yang terendah dan tertinggi dari seluruhnya
4. Carilah seluruh karyawan yang memiliki huruf ‘n’ dari nama depan maupun nama belakangnya
5. Gunakan regexp untuk mencari seluruh karyawan yang memiliki nama depan dengan Panjang 7 karakter dan huruf depannya ‘B’

Praktikum #9

Pengolahan Multi Tabel

Standar Kompetensi	
Mahasiswa dapat melakukan pengolahan data dengan menggunakan multi tabel	

Kompetensi Dasar	Indikator
1. Mampu menghubungkan dua tabel untuk menampilkan informasi	a. Mampu menggunakan INNER JOIN, STRAIGHT JOIN, RIGHT JOIN, dan LEFT JOIN untuk menghubungkan dua tabel. b. Mampu membedakan cara penggunaan INNER JOIN, STRAIGHT JOIN, RIGHT JOIN, dan LEFT JOIN
2. Mampu menghubungkan tiga tabel untuk menampilkan informasi	Mampu memperluas penggunaan INNER JOIN (yang secara optional digunakan oleh MySQL untuk JOIN), untuk pengolahan dengan menggunakan tiga tabel masukan.

A. Pengolahan dengan dua tabel atau lebih

Untuk memenuhi kebutuhan informasi yang lengkap, sering kali data tidak dapat diperoleh hanya dari satu tabel. Data yang diperoleh dari isian berbagai field harus diambil dari dua tabel atau lebih. Misal dengan menggunakan tabel **ruang**, **barang**, dan **isiruang** yang sudah dihasilkan dari praktikum sebelumnya, diinginkan informasi tentang barang-barang apa saja dan jumlahnya berapa yang ada pada suatu ruang, maka keinginan tersebut akan dapat dipenuhi dengan menghubungkan tabel **isiruang** dengan tabel **barang**. Pertanyaannya adalah kenapa?, karena data tentang nomor ruang, nomor barang dan jumlah barang ada di tabel **isiruang**, sedangkan data tentang nama barang ada di tabel **barang**.

MySQL dilengkapi dengan fasilitas untuk menghubungkan dua tabel atau lebih melalui suatu proses yang disebut JOIN. Untuk dapat dihubungkan, dua tabel harus mempunyai kolom atau field bersama, yaitu kolom yang mempunyai domain yang sama. Isi kolom bersama dari kedua tabel isinya akan dibandingkan atau dicocokkan untuk menentukan baris atau baris-baris dari dua tabel yang mempunyai hubungan.

Untuk proses JOIN yang diperlukan adalah:

- Atribut-atribut yang akan ditampilkan. Untuk memperjelas atribut mana yang dimaksud, maka penulisan nama atribut diawali dengan nama tabel dengan penghubung berupa tanda titik. Contoh: **isiruang.kodebarang**, untuk atribut kodebarang dari tabel **isiruang**
- Tabel-tabel yang digunakan sebagai sumber data
- Kondisi yang berisi kondisi atau kondisi-kondisi yang harus dipenuhi

Dengan proses JOIN tabel-tabel yang digunakan sebagai sumber data tidak akan mengalami pengubahan isi, jadi sebatas hanya digunakan sebagai input.

Macam hasil penggabungan untuk dua tabel ialah:

1. Hanya menampilkan data yang ditemukan pasangannya di semua tabel sumber.
2. Menampilkan data dengan acuan tabel sumber pertama, jika di tabel sumber kedua data tidak ditemukan, diisi dengan NULL.
3. Menampilkan data dengan acuan tabel sumber kedua, jika di tabel sumber pertama data tidak ditemukan, diisi dengan NULL.

Secara optional untuk proses JOIN digunakan macam yang pertama. Untuk data masukan yang terdiri lebih dari dua tabel, digunakan pengembangan dari proses dengan dua tabel.

B. Pengolahan dengan menggunakan dua tabel.

Untuk mempermudah pemberian contoh, ambil kembali tabel-tabel **ruang**, **barang**, dan **isiruang**.

1. Perintah INNER JOIN

Bentuk umum:

```
SELECT field1, field2, ....  
FROM tabel1 INNER JOIN tabel2  
ON kondisi
```

Penulisan nama field yang lengkap ialah dengan menambahkan nama tabel yang terkait, dengan diberi tanda penghubung berupa tanda titik (.). Contoh: **isiruang.noruang**, yang artinya field noruang dari tabel **isiruang**.

Dengan INNER JOIN akan dihasilkan baris-baris yang mempunyai hubungan diantara tabel1 dan tabel2, untuk yang tidak mempunyai hubungan tidak akan ditampilkan.

Jika diinginkan ditampilkan nomor ruang (dari tabel **isiruang**), dan fungsi ruang (dari tabel **ruang**), maka perintah dan hasilnya adalah sebagai berikut:

```
mysql> select isiruang.noruang, ruang.fungs i  
-> from isiruang inner join ruang  
-> where isiruang.noruang=ruang.noruang;  
+-----+-----+  
| noruang | fungs i |  
+-----+-----+  
| A101    | Ruang Seminar |  
| A102    | Ruang Tamu   |  
| A102    | Ruang Tamu   |  
| A201    | Lab Komputer |  
| B101    | Ruang Kuliah |  
| B303    | Ruang Kuliah |  
+-----+-----+  
21 rows in set (0.48 sec)
```

Atau karena INNER JOIN adalah optional, maka untuk keperluan di atas dapat juga digunakan perintah dengan tidak menyertakan INNER JOIN sebagai berikut:

```
mysql> select isiruang.noruang, ruang.fungs i  
-> from isiruang, ruang  
-> where isiruang.noruang=ruang.noruang;  
+-----+-----+  
| noruang | fungs i |  
+-----+-----+  
| A101    | Ruang Seminar |  
| A102    | Ruang Tamu   |  
| A102    | Ruang Tamu   |  
| A201    | Lab Komputer |  
| B101    | Ruang Kuliah |  
| B303    | Ruang Kuliah |  
+-----+-----+  
21 rows in set (0.00 sec)
```

Hasilnya ternyata sama.

2. Perintah STRAIGHT JOIN

Bentuk umum:

```

SELECT field1, field2, ....
FROM tabel1 STRAIGHT JOIN tabel2

```

Perhatikan untuk STRAIGHT JOIN tidak mengenal adanya klause where. Hasil operasi dari STRAIGHT JOIN ialah munculnya pasangan data yang ada pada kedua tabel. Sehingga jumlah record yang dihasilkan merupakan hasil perkalian dari jumlah record data tabel masukan pertama dengan jumlah record data tabel masukan kedua.

Misal diberikan perintah sebagai berikut.

```

mysql> select *
-> from isiruang straight join ruang;

```

Hasilnya adalah setiap data dari tabel **isiruang** dipasangkan dengan setiap data dari tabel **ruang**. Tampilan lengkap cukup panjang (189 baris), dan bagian akhir dari tampilan baris-baris yang dihasilkan adalah sebagai berikut

B303	409001	1	B303	B	3	Lab Komputer
B303	409001	1	B303	B	1	Ruang Kuliah
B303	409003	2	A101	A	1	Ruang Seminar
B303	409003	2	A102	A	1	Ruang Tamu
B303	409003	2	A201	A	2	Lab Komputer
B303	409003	2	B101	B	1	Ruang Kuliah
B303	409003	2	B102	B	1	Ruang Kuliah
B303	409003	2	B201	B	2	Ruang Kuliah
B303	409003	2	B202	B	2	Ruang Kuliah
B303	409003	2	B203	B	3	Lab Komputer
B303	409003	2	B303	B	3	Ruang Kuliah

189 rows in set (0.00 sec)

Perhatikan kembali jumlah baris yang ditampilkan, yaitu 189.

Dari mana angka tersebut dihasilkan?

3. Perintah RIGHT JOIN

Bentuk umum:

```

SELECT field1, field2, ....
FROM tabel1 RIGHT JOIN tabel2
ON kondisi

```

Dengan RIGHT JOIN maka yang dipakai sebagai dasar di dalam menampilkan hasil proses JOIN adalah tabel yang ditulis di sebelah kanan perintah JOIN, yaitu tabel2. Dengan demikian semua record dari tabel2 akan ditampilkan, dan untuk yang tidak mempunyai hubungan di tabel1 isi field yang berasal dari tabel1 diberi NULL.

Misal diminta menampilkan nomor ruang (dari tabel **isiruang**), fungsi ruang (dari tabel **ruang**), dan nomor barang (dari tabel **isiruang**), maka perintah dan hasil tampilannya adalah sebagai berikut.

```

mysql> select isiruang.noruang, ruang.fungsi, isiruang.nobarang
-> from isiruang right join ruang
-> on isiruang.noruang=ruang.noruang;
+-----+-----+-----+
| noruang | fungsi | nobarang |
+-----+-----+-----+
| A101    | Ruang Seminar | 110001
| A101    | Ruang Seminar | 210001
| A101    | Ruang Seminar | 309001
| A101    | Ruang Seminar | 409001
| A102    | Ruang Tamu   | 111001
| A102    | Ruang Tamu   | 211001
| A201    | Lab Komputer | 108001
| A201    | Lab Komputer | 109001
| A201    | Lab Komputer | 209001
| A201    | Lab Komputer | 308001
| A201    | Lab Komputer | 409002
| B101    | Ruang Kuliah | 108001
| B101    | Ruang Kuliah | 208001
| B101    | Ruang Kuliah | 308001
| B101    | Ruang Kuliah | 409001
| B101    | Ruang Kuliah | 409003
| NULL    | Ruang Kuliah | NULL
| NULL    | Ruang Kuliah | NULL
| NULL    | Ruang Kuliah | NULL
| NULL    | Lab Komputer | NULL
| B303    | Ruang Kuliah | 108001
| B303    | Ruang Kuliah | 208001
| B303    | Ruang Kuliah | 308001
| B303    | Ruang Kuliah | 409001
| B303    | Ruang Kuliah | 409003
+-----+-----+-----+
25 rows in set (0.00 sec)

```

Bagaimana dengan perintah dan hasil yang berikut ini?

```

mysql> select isiruang.noruang, ruang.fungsi, isiruang.nobarang
-> from ruang right join isiruang
-> on ruang.noruang=isiruang.noruang;
+-----+-----+-----+
| noruang | fungsi | nobarang |
+-----+-----+-----+
| A101    | Ruang Seminar | 110001
| A101    | Ruang Seminar | 210001
| A101    | Ruang Seminar | 309001
| A101    | Ruang Seminar | 409001
| A102    | Ruang Tamu   | 111001
| A102    | Ruang Tamu   | 211001
| A201    | Lab Komputer | 108001
| A201    | Lab Komputer | 109001
| A201    | Lab Komputer | 209001
| A201    | Lab Komputer | 308001
| A201    | Lab Komputer | 409002
| B101    | Ruang Kuliah | 108001
| B101    | Ruang Kuliah | 208001
| B101    | Ruang Kuliah | 308001
| B101    | Ruang Kuliah | 409001
| B101    | Ruang Kuliah | 409003
| B303    | Ruang Kuliah | 108001
| B303    | Ruang Kuliah | 208001
| B303    | Ruang Kuliah | 308001
| B303    | Ruang Kuliah | 409001
| B303    | Ruang Kuliah | 409003
+-----+-----+-----+
21 rows in set (0.00 sec)

```

Tampilan yang terakhir diperoleh dengan melakukan saling tukar tabel yang digunakan sebagai tabel1 dan tabel2 pada pengolahan sebelumnya. Amati hasilnya, dan bandingkan dengan hasil pengolahan sebelumnya. Seperti apa kesimpulan hasil pembandingan Saudara?

4. Perintah LEFT JOIN

Bentuk umum:

```

SELECT field1, field2, ....
FROM tabel1 LEFT JOIN tabel2
ON kondisi

```

Dengan LEFT JOIN aturan yang dipakai ialah kebalikan dari RIGHT JOIN.

Misal diminta menampilkan nomor ruang (dari tabel **isiruang**), nama barang (dari tabel **barang**), dan jumlah barang (dari tabel **isiruang**), maka perintah dan hasilnya adalah sebagai berikut:

```

mysql> select isiruang.noruang, barang.namabarang, isiruang.jumlah
-> from isiruang left join barang
-> on isiruang.nobarang=barang.nobarang;
+-----+-----+-----+
| noruang | namabarang | jumlah |
+-----+-----+-----+
| A101    | Meja Sidang      | 6       |
| A101    | Kursi Sidang     | 12      |
| A101    | LCD              | 1       |
| A101    | Layar Peraga     | 1       |
| A102    | Meja Tamu        | 1       |
| A102    | Kursi Tamu       | 4       |
| A201    | Meja Kuliah      | 2       |
| A201    | Meja Komputer    | 40      |
| A201    | Kursi Lipat       | 42      |
| A201    | LCD              | 1       |
| A201    | Layar Peraga     | 1       |
| B101    | Meja Kuliah      | 1       |
| B101    | Kursi Kuliah     | 50      |
| B101    | LCD              | 1       |
| B101    | Layar Peraga     | 1       |
| B101    | Whiteboard        | 2       |
| B303    | Meja Kuliah      | 1       |
| B303    | Kursi Kuliah     | 50      |
| B303    | LCD              | 1       |
| B303    | Layar Peraga     | 1       |
| B303    | Whiteboard        | 2       |
+-----+-----+-----+
21 rows in set (0.00 sec)

```

Bandingkan hasilnya dengan jika penempatan tabel yang digunakan sebagai sumber dibalik, yaitu tabel **barang** sebagai tabel1, dan tabel **isibarang** sebagai tabel2.

Perintah dan hasilnya ialah sebagai berikut.

```

mysql> select isiruang.noruang, barang.namabarang, isiruang.jumlah
-> from barang left join isiruang
-> on barang.nobarang=isiruang.nobarang;
+-----+-----+-----+
| noruang | namabarang | jumlah |
+-----+-----+-----+
| A201    | Meja Kuliah      | 2       |
| B101    | Meja Kuliah      | 1       |
| B303    | Meja Kuliah      | 1       |
| A201    | Meja Komputer    | 40      |
| A101    | Meja Sidang      | 6       |
| A102    | Meja Tamu        | 1       |
| B101    | Kursi Kuliah     | 50      |
| B303    | Kursi Kuliah     | 50      |
| NULL    | Kursi Kuliah     | NULL    |
| A201    | Kursi Lipat       | 42      |
| A101    | Kursi Sidang     | 12      |
| NULL    | Kursi Kuliah     | NULL    |
| A102    | Kursi Tamu        | 4       |
| NULL    | Kursi Kuliah     | NULL    |
| A201    | LCD              | 1       |
| B101    | LCD              | 1       |
| B303    | LCD              | 1       |
| A101    | LCD              | 1       |
| A101    | Layar Peraga     | 1       |
| B101    | Layar Peraga     | 1       |
| B303    | Layar Peraga     | 1       |
| A201    | Layar Peraga     | 1       |
| B101    | Whiteboard        | 2       |
| B303    | Whiteboard        | 2       |
| NULL    | Komputer Desktop | NULL   |
| NULL    | Printer Laser     | NULL   |
+-----+-----+-----+
26 rows in set (0.00 sec)

```

Apa yang berbeda dengan hasil sebelumnya?

C. Pengolahan dengan menggunakan tiga tabel.

Sebagai perluasan dari proses JOIN dengan menggunakan dua tabel, ialah proses JOIN dengan menggunakan tiga tabel.

Ambil kembali tabel-tabel **ruang**, **barang**, dan **isiruang**. Misal diminta ditampilkan sekaligus field-field sebagai berikut:

- nomor ruang dan jumlah barang : dari tabel **isiruang**
- fungsi ruang : dari tabel **ruang**, dan
- nama barang : dari tabel **barang**.

Perintah yang diberikan dan tampilan yang dihasilkan adalah sebagai berikut.

```

mysql> select isiruang.noruang,ruang.fungsi,barang.namabarang,isiruang.jumlah
-> from isiruang, ruang, barang
-> where isiruang.noruang=ruang.noruang
-> and isiruang.nobarang=barang.nobarang;
+-----+-----+-----+-----+
| noruang | fungsi | namabarang | jumlah |
+-----+-----+-----+-----+
| A101 | Ruang Seminar | Meja Sidang | 6 |
| A101 | Ruang Seminar | Kursi Sidang | 12 |
| A101 | Ruang Seminar | LCD | 1 |
| A101 | Ruang Seminar | Layar Peraga | 1 |
| A102 | Ruang Tamu | Meja Tamu | 1 |
| A102 | Ruang Tamu | Kursi Tamu | 4 |
| A201 | Lab Komputer | Meja Kuliah | 2 |
| A201 | Lab Komputer | Meja Komputer | 40 |
| A201 | Lab Komputer | Kursi Lipat | 42 |
| A201 | Lab Komputer | LCD | 1 |
| A201 | Lab Komputer | Layar Peraga | 1 |
| B101 | Ruang Kuliah | Meja Kuliah | 1 |
| B101 | Ruang Kuliah | Kursi Kuliah | 50 |
| B101 | Ruang Kuliah | LCD | 1 |
| B101 | Ruang Kuliah | Layar Peraga | 1 |
| B101 | Ruang Kuliah | Whiteboard | 2 |
| B303 | Ruang Kuliah | Meja Kuliah | 1 |
| B303 | Ruang Kuliah | Kursi Kuliah | 50 |
| B303 | Ruang Kuliah | LCD | 1 |
| B303 | Ruang Kuliah | Layar Peraga | 1 |
| B303 | Ruang Kuliah | Whiteboard | 2 |
+-----+-----+-----+-----+
21 rows in set (0.00 sec)

```

Dengan perintah di atas akan dhasilkan tampilan sebagaimana jika digunakan INNER JOIN, karena secara optional yang digunakan untuk proses JOIN adalah INNER JOIN.

Tugas A

Dengan menggunakan tabel-tabel **ruang**, **barang**, dan **isiruang** yang telah dihasilkan pada praktikum-praktikum sebelumnya.

Susun perintah untuk pengolahan-pengolahan sebagai berikut:

1. Menampilkan nomor ruang, fungsi ruang, dan jumlah barang untuk ruang-ruang yang terdapat LCD di dalamnya.
2. Menampilkan nomor ruang, fungsi ruang, jumlah barang untuk ruang kuliah yang mempunyai kursi dengan jumlah minimum 50.
3. Menampilkan nomor barang, nama barang, nomor ruang, dan fungsi ruang untuk ruang kuliah yang ada di lantai 2.
4. Menampilkan nomor barang, nama barang, jumlah barang, nomor ruang, dan fungsi ruang untuk barang-barang yang ada di Lab Komputer.
5. Menampilkan nomor barang, nama barang, jumlah barang, nomor ruang, dan fungsi ruang untuk barang-barang yang berupa alat peraga/whiteboard.

Tugas B : Menggunakan DB Organisasi, buatlah masing-masing 1 contoh join (INNER JOIN, STRAIGHT JOIN, RIGHT JOIN, dan LEFT JOIN) dari dua atau tiga tabel kemudian jelaskan query yang anda lakukan !

-oOo-