UKDW
Universitas Kristen Duta Wacana
YOGYAKARTA

Obedience
Integrity
Excellence
Service

Relational Database Model

Dr. **Rosa Delima**, S.Kom., M.Kom.
Lukas Chrisantyo, S.Kom., M.Eng.
Agata Filiana, S.Kom., M. Sc.
Maria Nila Anggia Rini, S.T., M.T.I.

Small Group Discussion #3

- Dari posisi terakhir kemarin, lanjutkan untuk menambahkan satu entitas lain yang ada hubungannya dengan entitas yang dikerjakan.
- Harus bisa menjelaskan hubungannya dengan kalimat:
  "Satu [entitas A] [kata kerja relasi aktif] dengan [satu/lebih dari satu] [entitas B];
  dan satu [entitas B] [kata kerja relasi pasif] dengan [satu/lebih dari satu] [entitas A]
- Khusus yang dapat entitas **matakuliah** cari entitas lain selain **mahasiswa**.

PRODUCTS	EMPLOYEES	MOVIES	MATAKULIAH	CARS
(Toko Cat)	(Rental Bus)	(IMDb)	(Akademik)	(Dealer Mobil)

SPAREPARTS	STORES	BUKU	MENU	PASIEN
(Bengkel)	(KFC Chains)	(Perpustakaan)	(Restoran)	(Klinik)

- Tambahkan di GSheet yang terakhir kemarin, lengkap dengan 5 atribut dan up to 9 instans.

Struktur Basis Data Relasional

Tabel **dosen**

nip	*namaPegawai*	*namaProdi*	*gajiTahunan*
10101	Sriyanto	Informatika	66000000
12121	Wahyu	Akuntansi	75000000
15151	Mozart	Sistem Informasi	68000000
22222	Einstein	Desain Produk	60000000
32343	Spielberg	Arsitektur	62000000
33456	Lucas	Kedokteran	95000000
44565	Katz	Biologi	79000000
58583	Charles	Informatika	83000000
76543	Simon	Biologi	85000000
76766	Peter	Sistem Informasi	75000000
83821	Bagong	Manajemen	66000000
98345	Karyono	Pendidikan Bahasa	76000000

- Sebuah relational database terdiri dari sekumpulan **tabel.**
- Masing-masingnya diberikan nama yang **unik.**
- Perhatikan tabel **dosen** berikut.
- Tabel ini memiliki empat kolom atribut.
- Tiap baris menunjukan kejadian antar atribut yang saling berhubungan.

Struktur Basis Data Relasional

- Demikian pula untuk tabel **matakuliah** di sini memiliki empat kolom juga.
- Perhatikan bahwa tiap dosen diidentifikasi berdasarkan *nip*, sedangkan mata kuliah diidentifikasi berdasarkan *idMtk.*
- Tabel ini memiliki empat kolom atribut.
- Tiap baris menunjukan kejadian antar atribut yang saling berhubungan.

Tabel **matakuliah**

idMtk	*namaMtk*	*namaProdi*	*sks*
BIO-101	Pengantar Biologi	Biologi	4
BIO-301	Genetika	Biologi	4
BIO-399	Biologi Komputasional	Biologi	3
TI-101	Teknologi Komputer	Informatika	4
TI-190	Game Design	Informatika	4
TI-315	Sistem Basis Data	Informatika	3
TI-319	Pemrograman Web	Informatika	3
DP-322	Estetika Produk	Desain Produk	3
AK-181	Perpajakan	Akuntansi	3
AR-351	Analisis Struktur I	Arsitektur	3
KD-200	Anatomi I	Kedokteran	3
PB-255	Grammar	Pendidikan Bahasa	4

Struktur Basis Data Relasional

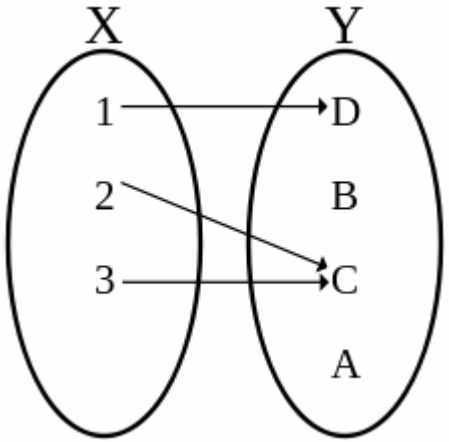
Tabel **prasyarat**

idMtk	*syaratMtk*
BIO-301	BIO-101
BIO-399	BIO-101
TI-190	TI-101
TI-315	TI-101
TI-319	TI-190

- Bagaimana dengan tabel **prasyarat** ini?
- Tabel ini menyimpan syarat mata kuliah untuk mata kuliah lain.
- Dari sini terlihat bahwa tabel **prasyarat** ini berisi dua mata kuliah yang saling berelasi, yaitu mata kuliah dan prasyaratnya.

Struktur Basis Data Relasional

- Secara umum, satu baris dalam sebuah tabel merepresentasikan sebuah *relationship* atau keterhubungan di antara sekumpulan nilai.
- Dari sini terlihat kembali hubungan dari konsep yang disampaikan E.F. Codd, yaitu tabel dengan konsep matematika dari *relasi*.
- Dalam terminology matematika, *tuple* adalah sebuah sekuens atau serangkaian nilai.



- Tuple dengan *n* nilai, nantinya akan kita sebut satu baris tabel dengan *n* atribut.

Struktur Basis Data Relasional

- Ingat kembali istilah entity. Tabel yang dicontohkan di atas adalah sekumpulan instans entitas **dosen**, **mata kuliah**, dan **prasyarat** mata kuliah.
- Entitas **dosen** memiliki atribut *nip, namaPegawai, namaProdi,* dan *gajiTahunan.*
- Entitas **mata kuliah** memiliki atribut *idMtk, namaMtk, namaProdi,* dan *sks.*
- Untuk setiap atribut di tabel, ada sekumpulan nilai yang dibolehkan, yang disebut **domain** dari atribut. Contoh: *gajiTahunan* dan *sks* berupa angka dengan besaran tertentu.

Struktur Basis Data Relasional – Atomic

- Menurut Codd, untuk setiap relasi *r*, domain dari seluruh atribut dari relasi *r* harus **atomic**.
- Domain disebut atomic jika elemen dari setiap domain merupakan unit yang tidak bisa dibagi-bagi lagi.
- Contoh jika sebuah mata kuliah memiliki kelas paralel A, B, C, D, maka atribut kelas paralel ini **tidak atomic**.

Struktur Basis Data Relasional – Null

- Nilai **null** adalah suatu nilai khusus yang menjelaskan bahwa untuk sebuah atribut dari satu baris, tidak memiliki nilai sama sekali.
- Contoh seorang dosen kemungkinan belum memiliki Scopus ID. Maka misalkan ada atribut *scopusID*, maka atribut itu dapat dibiarkan kosong.
- Kosong = Null ≠ Nol
- Kita akan mendapati nantinya bahwa nilai null akan menimbulkan beberapa kerepotan ketika hendak mengakses dan mengupdate database.

Relation Schema

- Dalam penulisannya, tabel dapat direpresentasikan sebagai bentuk *relation schema* yang lebih sederhana:
  - *dosen (nip, namaPegawai, namaProdi, gajiTahunan)*
  - *matakuliah (idMtk, namaMtk, namaProdi, sks)*
  - *prasyarat (idMtk, syaratMtk)*

- Perhatikan juga tabel **prodi** dengan relation schema *prodi (namaProdi, gedung, kaprodi)*

- Ada satu kolom atribut yang di tabel-tabel sebelumnya juga sering muncul. Apakah itu?

Tabel **prodi**

namaProdi	*gedung*	*kaprodi*
Biologi	Euonia	Dhira Satwika
Arsitektur	Chara	Sita Amijaya
Desain Produk	Chara	Kristian Oentoro
Akuntansi	Didaktos	Christine Novita
Manajemen	Didaktos	Sisnuhadi
Pendidikan Bahasa	Euonia	Lemmuela Alvita
Kedokteran	Makarios	Ida Ayu Triastuti
Informatika	Agape	Gloria Virginia
Sistem Informasi	Agape	Jong Jek Siang

Relation Schema

- Atribut *namaProdi* muncul di schema **dosen, matakuliah** dan **prodi**.
- Duplikasi ini bukan suatu ketidaksengajaan, namun dimaksudkan sebagai salah satu cara merelasikan tuple-tuple dari relasi yang berbeda.
- Contoh, jika kita membutuhkan informasi *"Siapa dosen yang berkantor di gedung Agape?"*
- Ada yang bisa menjelaskan langkah-langkah mendapatkan jawabannya?

Relation Schema

- Tiap mata kuliah bisa ditawarkan beberapa kali baik di satu semester yang sama maupun antar semester.
- Kita membutuhkan suatu relasi untuk mendeskripsikan peristiwa tersebut, misalnya dengan schema: *mtkDitawarkan (idMtk, grup, semester, tahun, gedung, ruang, kodeSesi)*

Tabel **mtkDitawarkan**

idMtk	*grup*	*semester*	*tahun*	*gedung*	*nomorRuang*	*kodeSesi*
BIO-101	A	Gasal	2022	Biblos	B.3.1	A1
BIO-301	A	Gasal	2022	Biblos	B.3.1	A2
BIO-301	B	Gasal	2022	Biblos	B.3.2	A2
BIO-399	A	Genap	2021	Didaktos	D.1.1	B3
TI-101	A	Genap	2021	Didaktos	D.1.2	A2
TI-190	A	Gasal	2022	Didaktos	D.3.1	C2
TI-190	B	Gasal	2022	Didaktos	D.3.3	C2
TI-190	C	Gasal	2022	Hagios	H.2.1	C2
TI-315	A	Gasal	2021	Hagios	H.2.1	D1
TI-319	A	Genap	2021	Biblos	B.3.1	D1
DP-322	A	Gasal	2021	Biblos	B.3.1	D1
AK-181	A	Gasal	2022	Chara	C.3.2	D2
AK-181	A	Genap	2021	Chara	C.3.3	D2
AR-351	A	Gasal	2022	Chara	C.3.7	E3
KD-200	A	Genap	2021	Koinonia	K.1.1	E1
PB-255	A	Genap	2022	Koinonia	K.1.1	E2

Relation Schema

- Berikutnya, kita membutuhkan sebuah relasi yang mendeskripsikan asosiasi antara dosen dengan mata kuliah yang ditawarkan.
- Schema:
  pengajaran (nip, idMtk, grup, semester, tahun)

Tabel **pengajaran**

nip	*idMtk*	*grup*	*semester*	*tahun*
10101	TI-101	A	Gasal	2022
10101	TI-315	A	Gasal	2022
10101	TI-101	B	Genap	2021
12121	AK-181	A	Gasal	2022
32343	AR-351	C	Gasal	2022
22222	DP-322	A	Genap	2021
22222	DP-322	C	Gasal	2022
44565	BIO-101	B	Genap	2021
44565	BIO-101	A	Gasal	2022
44565	BIO-101	C	Gasal	2022
33456	KD-200	B	Gasal	2022
33456	KD-200	C	Gasal	2022

Relation Schema

- Selanjutnya, kita bisa menambahkan relasi yang lain untuk melengkapi database akademik universitas.

- Dari relasi/tabel yang sudah ada: *dosen*, *prodi*, *matakuliah*, *mtkDitawarkan*, *prasyarat*, dan *pengajaran*, kita bisa menambahkan:
  - *mahasiswa (nim, nama, namaProdi, sksTotal)*
  - *registrasi (nim, idMtk, grup, semester, tahun, nilai)*
  - *ruangKelas (gedung, nomorRuang, kapasitas)*
  - *sesi (kodeSesi, hari, jamMulai, jamSelesai)*

Kunci

- Kita perlu memiliki cara untuk mencirikan bagaimana tuples di dalam suatu relasi bisa dibedakan satu dengan yang lainnya.
- Kebutuhan ini bisa dijawab dengan bantuan atribut.
- Atribut yang dipilih harus bisa mengidentifikasi secara unik tiap tuplenya.
- Dengan kata lain, tidak ada dua tuples di satu relasi yang sama, diperbolehkan memiliki nilai sama untuk seluruh atribut yang nantinya dinyatakan sebagai kunci.

Superkey

- Superkey adalah himpunan dari satu atribut atau lebih yang—ketika dilihat secara bersamaan—bisa mengidentifikasi secara unik suatu tuple di dalam relasi tersebut.
- Contoh, atribut *nip* dari relasi *dosen* sudah cukup untuk membedakan satu tuple dosen dengan yang lainnya. Sehingga, *nip* bisa dikatakan merupakan superkey.
- Atribut *namaPegawai* dari relasi *dosen* bukan merupakan superkey, karena beberapa dosen bisa jadi memiliki nama yang kembar.
- Katakan ***R*** adalah seluruh atribut dari relasi ***r***. Jika ***K*** yang merupakan subset dari ***R*** merupakan superkey bagi ***r***, maka tidak boleh ada dua tuples atau lebih yang memiliki nilai yang persis sama untuk setiap atribut ***K***.

Candidate Key

- Superkey bisa berisi atribut yang lainnya di luar yang sudah ditetapkan. Contoh, kombinasi *nip* dan *namaPegawai* adalah superkey untuk relasi *dosen.*
- Jika *K* adalah superkey, demikian juga dengan setiap superset dari *K.*
- Untuk superkey yang tidak ada lagi subset darinya yang bisa jadi superkey, dikatakan sebagai minimal superkey.
- Minimal superkey seperti ini dapat disebut sebagai **candidate key.**

Candidate Key

- Adalah dimungkinkan bahwa beberapa sekumpulan atribut yang berbeda bisa bertindak sebagai candidate key.
- Asumsikan ada kombinasi *namaPegawai* dan *namaProdi* cukup untuk membedakan setiap member dari relasi *dosen.* Maka baik {*nip*} dan {*namaPegawai, namaProdi*} adalah candidate key.
- Walaupun *nip* dan *namaPegawai* secara bersamaan bisa membedakan tuple-tuplenya *dosen*, namun gabungan keduanya {*nip, namaPegawai*} tidak membentuk candidate key, karena atribut *nip* sudah ditetapkan sebagai candidate key.

Primary Key

- Kita menggunakan istilah primary key (PK) untuk mewakili suatu candidate key yang telah dipilih oleh DB designer sebagai perangkat utama untuk pembedaan tuple di dalam suatu relasi.
- Penentuan suatu key merepresentasikan konstrain yang ada di dunia nyata yang tertuang di aturan bisnis yang akan dimodelkan.
- Secara konsensus, penulisan primary key diletakkan di atribut paling awal. Sedangkan di relation schema dibedakan dengan menambahkan underline.
- Contoh: *ruangKelas (gedung, nomorRuang, kapasitas)* → jika nomorRuang = 1, 2, 3 … Sedangkan untuk kasus kampus kita cukup: *ruangKelas (nomorRuang, gedung, kapasitas)*

Primary Key

- Penentuan PK harus hati-hati. PK dipilih dari suatu atribut yang nilainya tidak pernah atau sangat jarang sekali berubah.
- Jika memilih NIK untuk menjadi PK, bagaimana dengan WNA?
- Jika memilih NomorHP untuk menjadi PK, bagaimana jika usernya sering ganti nomor?
- Jika memilih Email untuk menjadi PK, bagaimana jika usernya banyak yang gaptek?
- Memilih auto increment number pasti aman, namun resikonya mudah ditebak.
- Berdasarkan definisinya, maka sifat PK: harus unik dan tidak boleh null.

Primary Key

Berikut ini sekumpulan relasi dari database akademik lengkap dengan PK-nya.

- *prodi (namaProdi, gedung, kaprodi)*
- *matakuliah (idMtk, namaMtk, namaProdi, sks)*
- *dosen (nip, namaPegawai, namaProdi, gajiTahunan)*
- *mtkDitawarkan (idMtk, grup, semester, tahun, gedung, nomorRuang, kodeSesi)*
- *pengajaran (nip, idMtk, grup, semester, tahun)*
- *mahasiswa (nim, nama, namaProdi, sksTotal)*
- *registrasi (nim, idMtk, grup, semester, tahun, nilai)*
- *ruangKelas (nomorRuang, gedung, kapasitas)*
- *sesi (kodeSesi, hari, jamMulai, jamSelesai)*
- *prasyarat (idMtk, syaratMtk)*

Foreign Key

- Konstrain Foreign Key (FK) terhadap atribut-atribut *A* dari relasi r_1 pada PK *B* dari relasi r_2 menyatakan bahwa untuk setiap instans database, nilai dari *A* untuk setiap tuple di r_1 harus juga merupakan nilai *B* untuk beberapa tuple di r_2 .
- Sekumpulan atribut *A* disebut FK dari r_1, yang mengacu/me-reference ke r_2.
- Relasi r_1 juga disebut sebagai referencing relation (relasi yang mengacu) terhadap r_2 yang disebut sebagai referenced relation (relasi yang diacu).

Foreign Key

- Contoh:
  - *matakuliah (idMtk, namaMtk, namaProdi, sks)*
  - *dosen (nip, namaPegawai, namaProdi, gajiTahunan)*
  - *mahasiswa (nim, nama, namaProdi, sksTotal)*

- Atribut *namaProdi* mengacu ke *namaProdi* yang bertindak sebagai PK di relasi *prodi.*
  - *prodi (namaProdi, gedung, kaprodi)*

- Temukan lagi FK yang lain dari slide 21!

Foreign Key Referential Integrity

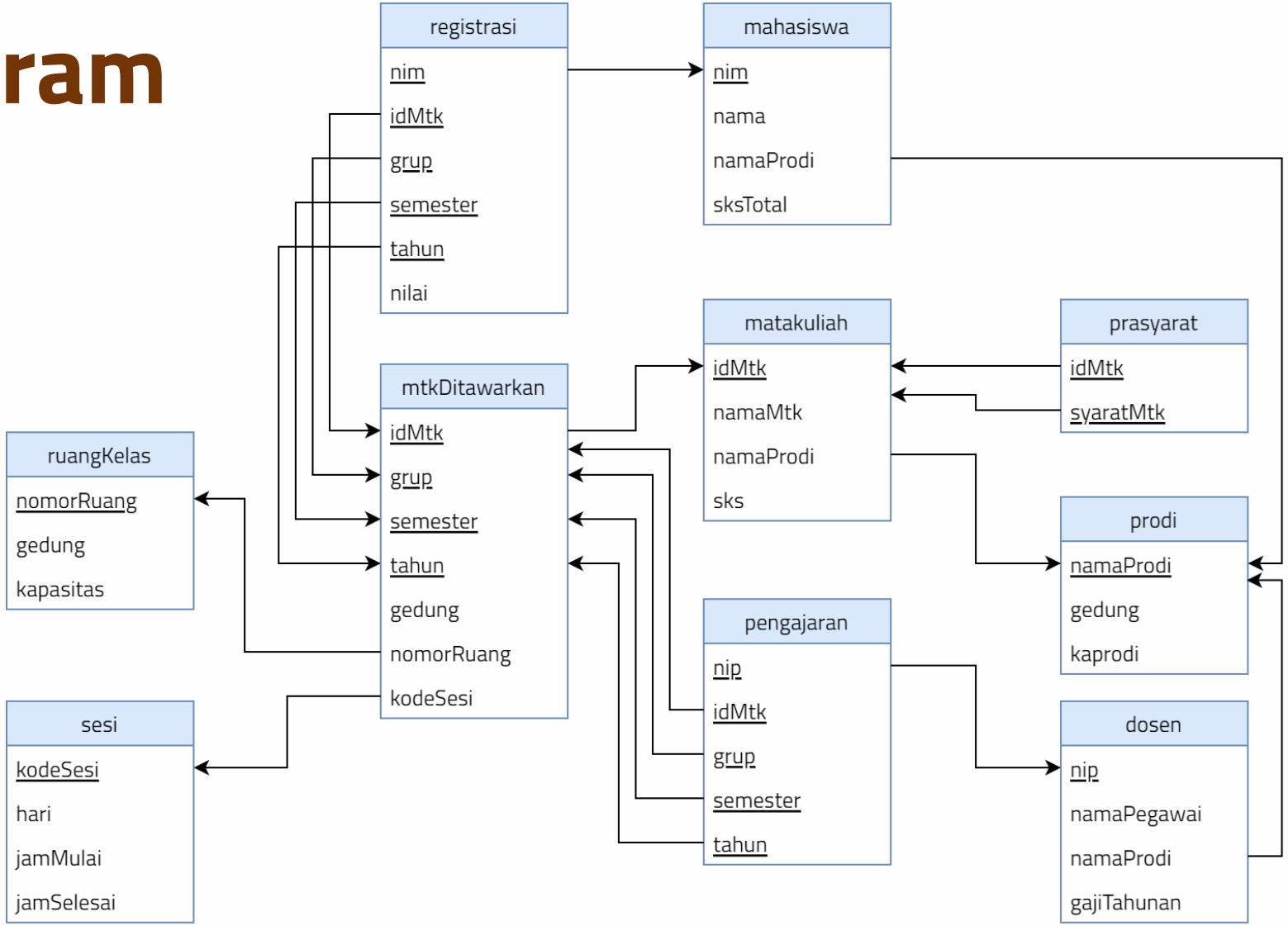
- Kita mungkin tidak bisa menjawab pertanyaan *"Siapa dosen yang berkantor di gedung Agape?"* jika misalnya ada satu tuple dosen yang berisi: `(81828, "Johnny", "Teologi", 80000000)`. Mengapa?
- Konstrain RI membutuhkan untuk nilai-nilai yang muncul di atribut FK di referencing relation, harus ada di referenced relation sebagai PK.
- Bagaimana jika di tabel *prodi,* nama prodi diubah ke "Teknik Informatika"? Boleh tidak?
- Bagaimana jika di tabel *prodi,* baris yang berisi "Biologi" dihapus? Boleh tidak?
- Maka inilah yang menjadi core dari konsep database relational: PK-FK.

Composite Key - Surrogate Key

- Kita cek ulang kasus contoh yang di luar kampus kita:
  ruangKelas (gedung, nomorRuang, kapasitas) → jika nomorRuang = 1, 2, 3 ...
- Ada dua atribut yang berlaku sebagai PK. Inilah yang disebut composite key. Lalu bagaimana FK-nya?
- Pengelolaan umumnya akan merepotkan jika PK berupa composite key.
- DB designer dapat "membangkitkan" sebuah kunci auto number untuk mempermudah pengacuan PK-FK.
- Inilah yang disebut surrogate key.

Schema Diagram

- Tanda panah menunjukkan alur referensi dari FK ke PK-nya.
- Kita pelajari lebih lanjut di pertemuan berikutnya (ER Diagram).



What questions do you have?