2401960435 – Reynaldy Sentosa – UAS DBD.

1a. Aljabar Relasional adalah operasi berbahasa prosedural tingkat tinggi yang dapat digunakan untuk membentuk/menghasilkan sebuah relasi atau tabel baru tanpa mengubah isi dari tabel aslinya. Kemudian, hasil/output yang berupa tabel baru tersebut dapat menjadi masukan/input pada operasi aljabar relasional lainnya. Dalam hal ini, operator dari operasi aljabar relasional terdiri atas operator unary dan operator biner. Operator unary dapat digunakan untuk mengoperasikan satu tabel, sedangkan operator biner dapat digunakan untuk mengoperasikan hubungan antartabel/antarrelasi. Operator unary itu misalnya operator selection (σ) dan projection (Π), sedangkan operator biner itu contohnya natural join (⋈) dan set difference (-). Operator selection/query dapat digunakan untuk menghasilkan sebuah tabel beserta keseluruhan tupel (baris datanya) yang (nilai dari atribut-atributnya) memenuhi kondisi yang ditentukan. Operator projection dapat digunakan untuk menghasilkan sebuah tabel yang bersubset vertikal, artinya tabel yang baru dihasilkan tersebut berisikan keseluruhan tupel dari atribut-atribut yang ditentukan. Jika baik kondisi yang ditentukan pada operator selection maupun operator projection diabaikan, maka hasil dari kedua operator tersebut ialah sebuah tabel yang lengkap dengan keseluruhan tupel dan atributnya. Di sisi lain, kita juga dapat menggunakan operator penugasan (<-) untuk memerintahkan agar sebuah variabel menampung (tabel baru dari) hasil operasi aljabar relasional lain sebagai nilainya (Connolly & Begg, 2015, hal. 168-170 dan 173).

R1 <- σ Lama\_inap > 2 (TAMU).

HasilAkhir <- Π Nama\_tamu, Id\_Tamu (R1).

1b. Kita dapat menggunakan operator rename (ρ) untuk memberi nama pada variabel penampung beserta atribut-atribut pada tabel yang ditampungnya. Selain itu, kita dapat menggunakan operator aggregate untuk mengagregasikan setiap nilai yang memiliki kesamaan atribut. Dalam hal ini, kita dapat menggunakan operator/fungsi MAX untuk memperoleh nilai terbesar/tertinggi pada sebuah atribut. Di sisi lain, kita juga dapat menggunakan natural join untuk penggabungan antartabel yang memiliki kesamaan atribut (baik jumlah, nama, maupun tipe datanya) atau bersifat union-compatible. Penggabungan/Join merupakan operasi cartesian product dengan kondisi tertentu (jika natural, maka kondisinya union-compatible tersebut). Hal ini di mana cartesian product (yang beroperator X) adalah operasi aljabar relasional yang membentuk sebuah tabel baru sebagai hasil dari pasangan setiap nilai tupel yang mungkin antara tabel satu dan tabel lainnya (Connolly & Begg, 2015, hal. 172, 174-175, dan 178-179).

R1 <- Π Id\_kamar, Jenis\_kamar, harga (KAMAR).

R2 <- MAX harga (KAMAR).

Cara 1:

* R3 <- ρ (Harga\_maks) (R2).
* R4 <- σ (R3).
* HasilAkhir <- σ Harga = R4 (R1).

Cara 2:

* R3 <- ρ (harga) (R2).
* R4 <- KAMAR ⋈ R3.
* HasilAkhir <- Π Id\_kamar, Jenis\_kamar, harga (R4).

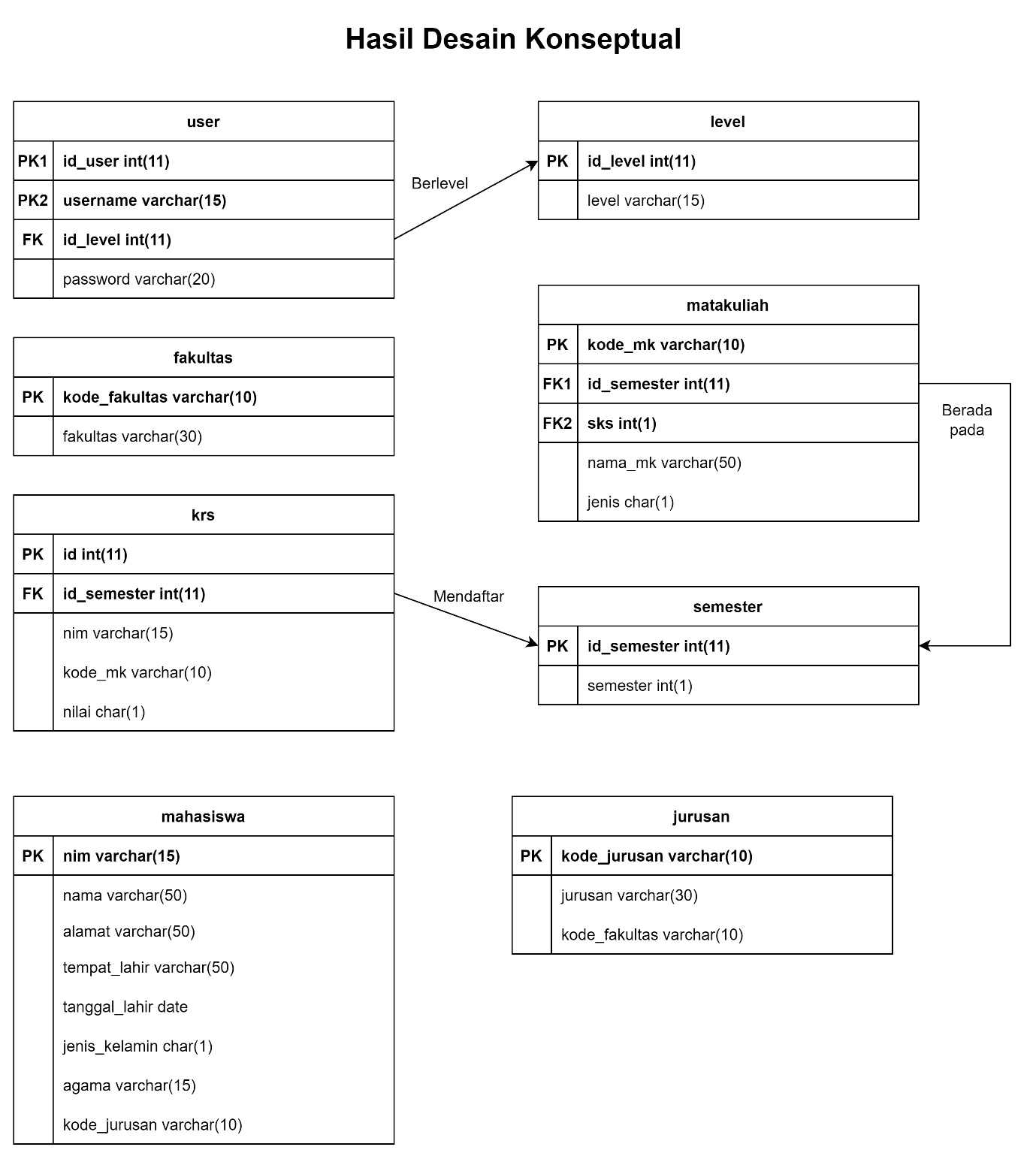
1c. Operator set difference dapat digunakan untuk menghasilkan sebuah tabel baru yang terbentuk dari pengecualian/perselisihan antartabel, misalnya antara Tabel R dan Tabel S. Pengecualian yang dimaksud berarti tabel baru tersebut harus mengandung keseluruhan nilai tupel Tabel R yang tidak berkemiripan dengan setiap nilai tupel Tabel S (Connolly & Begg, 2015, hal. 171).

R1 <- Π NIP (PEGAWAI) - Π NIP (TRANSAKSI\_MASUK) - Π NIP (TRANSAKSI\_KELUAR).

R2 <- PEGAWAI ⋈ R1.

HasilAkhir <- Π Nama\_pegawai, NIP (R2).

2a. Conceptual Model adalah tahapan awal Database Design pada Database System Development Lifecycle untuk menentukan data-data yang diperlukan berdasarkan persyaratan oleh klien. Selanjutnya, kita dapat menentukan atribut-atribut beserta relasinya. Jika dicermati, maka atribut ‘semester’ pada tabel “akad matakuliah” tampak kurang tepat, sehingga saya akan menulis ulang nama atribut tersebut menjadi ‘id\_semester’. Kita tidak perlu menentukan tipe-tipe entitas dan candidate key karena setiap tabel/entitas sudah memiliki nama dan primary key. Namun demikian, penulisan nama depan “akad ” pada setiap tabel terlihat tidak efektif, sehingga saya memutuskan untuk menghapusnya. Setelah itu, kita perlu menentukan tipe-tipe relasinya (Connolly & Begg, 2015, hal. 503, 508-511, dan 517).



2b. Logical Model adalah tahap lanjutan dari Conceptual Model untuk memastikan agar Entity Relationship Diagram (ERD) dari proses Database Design ini terstruktur dengan baik. Dengan demikian, kita dapat memulainya dengan memperbaiki relasi antartabel beserta multiciplity-nya menggunakan pendekatan Strong and Weak Entity. Jika dicermati, maka pemilihan atribut sks sebagai foreign key terasa tidak tepat karena tidak ada weak entity atau tabel selain matakuliah yang menggunakan/memiliki atribut sks. Selain itu, terdapat beberapa entitas (lemah) yang bergantung pada entitas lain. Dalam hal ini, baik tabel fakultas, tabel level, maupun tabel semester merupakan entitas kuat karena dapat berdiri sendiri dan mempengaruhi (atribut pada) entitas lemah. Dengan demikian, atribut pemengaruh pada entitas kuat dapat dijadikan primary key, sedangkan atribut terpengaruh pada entitas lemah dapat dijadikan foreign key. Selanjutnya, saya akan mengatur setiap atribut selain primary key menjadi mandatory (wajib bernilai) untuk menjaga keutuhan basis data (Connolly & Begg, 2015, hal. 528, 530-531, dan 540-541).

A picture containing letter

Description automatically generated

Selama proses desain, akhirnya saya menyadari bahwa tabel user dan tabel level belum terhubung dengan tabel lainnya. Saya pun mengambil inisiatif untuk menghubungkan tabel mahasiswa dengan tabel user karena seorang mahasiswa harus memiliki sebuah akun dan sebuah akun harus dimiliki oleh seorang mahasiswa. Dengan demikian, saya akan mengubah nama atribut username pada tabel user menjadi “nim” dan menjadikannya sebagai foreign key.

Text, letter

Description automatically generated

Proses Perancangan ERD: [Tautan Diagrams.net](https://drive.google.com/file/d/1xwLAdi4deMaJtogF4lbRbupX3Pc6SWXm/view?usp=sharing).

2c. Physical Model adalah tahap akhir setelah Logical Model untuk mengetahui bagaimana Database Management Systems (DBMS) yang kita pilih dapat mengimplementasikan ERD yang telah dirancang. Dalam hal ini, kita tidak perlu mendefinisikan base relations karena telah mengetahui tipe data dan (referential) integrity constraint pada setiap atribut. Kemudian, kita dapat menambahkan beberapa atribut turunan, misalnya atribut “jumlah\_jurusan” pada tabel fakultas, atribut “jumlah\_matakuliah” pada tabel semester, dan atribut “jumlah\_krs” pada tabel mahasiswa meskipun ketiganya belum terlalu dibutuhkan. Di sisi lain, kita juga belum memerlukan pengorganisasian file dan pemilihan indeks dengan mengurutkan nilai-nilai salah satu atribut pada suatu tabel (Connolly & Begg, 2015) (Connolly & Begg, 2015, hal. 562, 565-567, dan 574-575).

3a. Transaksi merupakan beberapa baris perintah SQL yang saling berkelanjutan dari baris pertama hingga baris terakhir. Transaksi dapat dimulai setelah perintah “BEGIN TRAN” dan tidak akan mempengaruhi transaksi lainnya (misalnya jika perintah “BEGIN TRAN” pertama dan kedua tidak sengaja terjalankan, maka transaksi kedua akan terabaikan secara otomatis). Akhirnya, transaksi dapat diselesaikan secara sukses dengan menjalankan keyword “COMMIT”. Namun, jika keyword “ROLLBACK” dijalankan sebelumnya, maka transaksi (yang mencakup berbagai perintah/perubahan) akan dibatalkan (Connolly & Begg, 2015, hal. 261).

--3a

BEGIN TRAN

INSERT INTO

PENERBIT

VALUES

(

'ER',

'Erlangga',

'Jakarta'

),

(

'YG',

'Yudhistira Galia',

'Jakarta'

),

(

'SJ',

'Sentosa Jaya',

'Bali'

),

(

'AG',

'Anggora',

'Surakarta'

)

INSERT INTO

PENGARANG

VALUES

(

'M-002',

'Akbar Tanjung'

),

(

'M-003',

'Titiek Suryaningsih'

),

(

'M-004',

'Reynaldy Sentosa'

),

(

'P-001',

'Ruslan'

)

ROLLBACK

COMMIT

--3b

BEGIN TRAN

UPDATE

PENERBIT

SET

Lokasi = 'Yogyakarta'

WHERE

Lokasi LIKE 'Yogya'

ROLLBACK

COMMIT

3c. Dalam hal ini, kita memerlukan proses inner atau natural join dengan kesamaan nilai atribut primary key pada setiap tabel (Connolly & Begg, 2015, hal. 219-220).

--3c

SELECT

Judul,

Nama\_Pengarang,

Nama\_Penerbit

FROM

BUKU1,

PENGARANG,

PENERBIT

WHERE

BUKU1.Kode\_Buku = PENGARANG.Kode\_Buku AND

BUKU1.Kode\_Penerbit = PENERBIT.Kode\_Penerbit

3d. Kita dapat menggunakan keyword “DISTINCT” untuk mencegah penampilan lebih dari satu data dari satu atribut atau lebih yang bernilai sama antartupelnya. Dengan demikian, antara tupel yang satu dan tupel-tupel lain akan menampilkan data/nilai atribut yang unik atau saling berbeda (Connolly & Begg, 2015, hal. 199-200).

--3d

SELECT

DISTINCT

Judul

FROM

BUKU1 b1 JOIN

BUKU2 b2 ON

b1.Kode\_Buku = b2.Kode\_Buku

WHERE

YEAR(Tgl\_Masuk) > 1997

3e. Dikarenakan Kode\_Rinci menyatakan kode untuk masing-masing buku dengan judul yang sama, maka nilai-nilai atributnya dapat digunakan sebagai indeks untuk menentukan jumlah dari setiap judul buku dan bukan edisinya(, misalnya ‘C1’ menyatakan buku pertama dan ‘C2’ menyatakan buku kedua). Dengan demikian, setiap judul buku yang memiliki nilai atribut Kode\_Rinci seperti ‘C2’, ‘C3’, dan seterusnya pasti berjumlah lebih dari satu buku.

--3e

SELECT

Judul,

Nama\_Pengarang,

Nama\_Penerbit

FROM

BUKU1 b1 JOIN

BUKU2 b2 ON

b1.Kode\_Buku = b2.Kode\_Buku JOIN

PENGARANG ON

b1.Kode\_Buku = PENGARANG.Kode\_Buku JOIN

PENERBIT ON

b1.Kode\_Penerbit = PENERBIT.Kode\_Penerbit

WHERE

Kode\_Rinci LIKE 'C2'

--3f

SELECT

Judul

FROM

BUKU1 b JOIN

PENERBIT p ON

b.Kode\_Penerbit = p.Kode\_Penerbit

WHERE

Edisi = 1 AND

Nama\_Penerbit LIKE 'Gramedia'

# **Referensi**

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