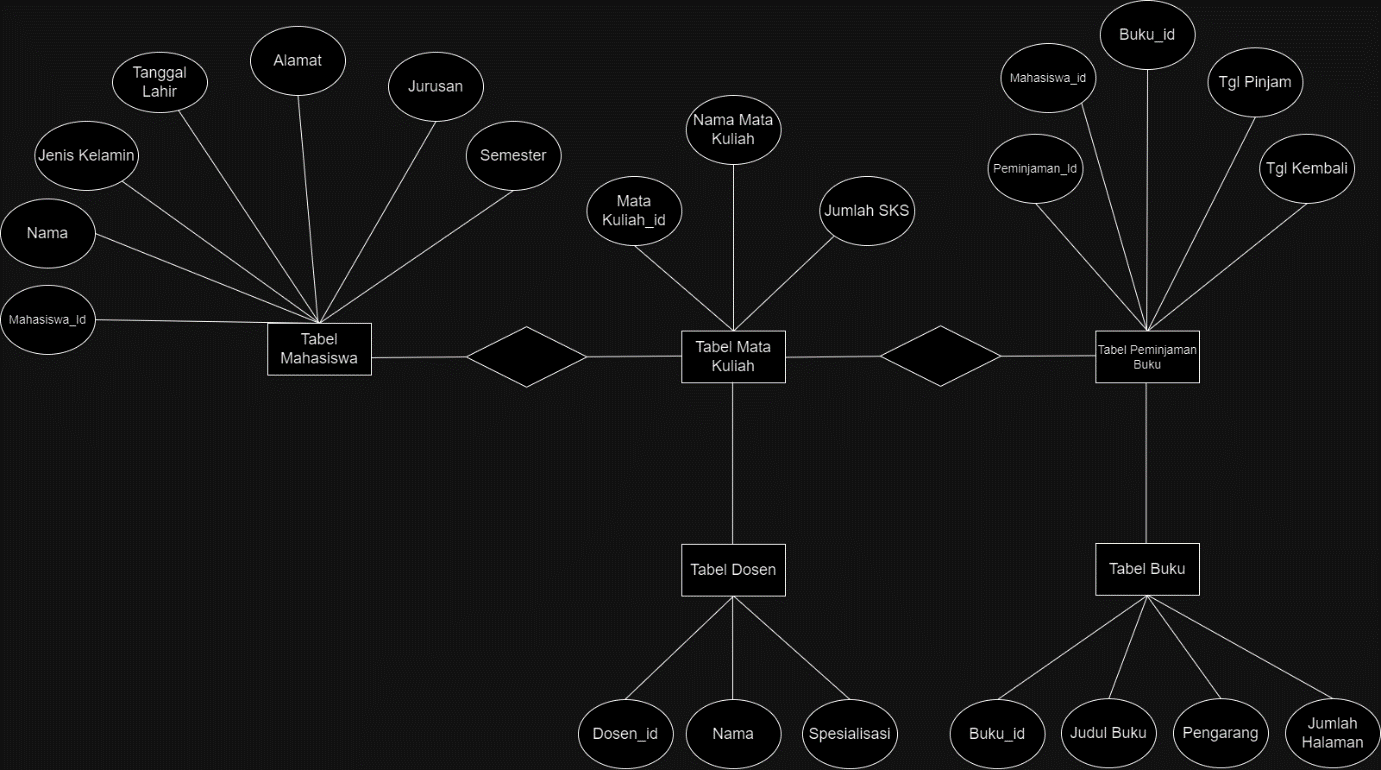
1. Diagram ERD



1. Model Relasional
2. Penjelasan detail :

* Atribut : Merupakan kolom dalam sebuah relasi yang mewakili karakteristik atau properti tertentu dari tupel. Atribut memiliki nama dan tipe data, seperti pada Tabel Mahasiswa (mahasiswa\_id, nama, jenis kelamin, tanggal lahir, alamat, jurusan, dan semester). Tabel Mata Kuliah (mata kuliah\_id, nama mata kuliah, dan jumlah sks), Tabel Dosen (dosen\_id, nama, dan spesialisasi). Tabel Peminjaman Buku (peminjaman\_id, mahasiswa\_id, buku\_id, tanggal pinjam, dan tanggal kembali), Tabel Buku (buku\_id, judul buku, pengarang, dan jumlah halaman).
* Tupel : Merupakan kumpulan nilai data yang mewakili satu baris dalam tabel. Tupel berisi nilai untuk semua atribut dalam tabel.
* Cardinality : Merupakan hubungan yang menunjukkan hubungan kuantitatif antara dua tabel.
* Domain : Merupakan kumpulan nilai yang valid untuk suatu atribut.
* Degree : Jumlah atribut dalam sebuah relasi.

1. Relasional Key

* Super Key : Merupakan kumpulan atribut dalam tabel yang bisa mengidentifikasi setiap baris secara unik. Ini bisa menjadi kandidat kunci, tapi mungkin bukan pilihan terbaik. Tabel bisa memiliki banyak superkey, seperti mahasiswa\_id, nama, jenis kelamin, tanggal lahir, alamat, dan jurusan dalam tabel mahasiswa yang mengidentifikasi setiap mahasiswa secara unik.
* Candidate Key : Merupakan superkey yang paling sederhana dan tidak mengandung atribut berlebihan. Ini adalah pengenal unik untuk setiap baris tanpa informasi yang tidak diperlukan. Tabel bisa memiliki beberapa kunci kandidat. Misalnya, dalam tabel siswa, jika "ID Siswa" mengidentifikasi setiap siswa secara unik, maka itu adalah kunci kandidat karena sederhana dan tidak memasukkan atribut "Email". Pada tabel yang di sebutkan, candidat key memiliki data unik sehingga seluruh atribut diperlukan untuk data informasi.
* Primary Key : Merupakan candidate key yang dipilih sebagai pengenal unik untuk setiap baris dalam tabel. Setiap tabel hanya punya satu kunci utama. Pemilihan kunci utama didasarkan pada faktor keunikan, sederhana, dan frekuensi penggunaan untuk menggabungkan tabel. Misalnya, dalam tabel mahasiswa, "NIM" bisa jadi pilihan bagus untuk kunci utama karena pendek, unik, dan sering digunakan untuk merujuk data mahasiswa di tabel lain.
* Alternate Key : Merupakan yang tidak dipilih sebagai primary key disebut alternate key. Meski juga bisa mengidentifikasi baris dalam tabel secara unik, alternate key bukanlah pengenal utama yang ditetapkan. Memiliki alternate bisa membantu dalam membuat indeks tambahan atau situasi di mana primary key bukan pilihan efisien untuk mencari data. Misalnya pada tabel dosen, memiliki primary key “nama” sehingga alternate key yang dapat digunakan adalah dengan menyebutkan salah satu candidat key “pemrograman”.
* Foreign Key : Merupakan grup kolom dalam suatu tabel yang merujuk ke primary key tabel lain disebut foreign key. Ini menghubungkan dua tabel, memungkinkan data di antara keduanya bisa dikaitkan dan ditanyakan. Foreign sangat penting untuk menjaga integritas data dan konsistensi dalam database relasional. Misalnya pada “tabel peminjaman buku” maka foreign dapat menghubungkan pada “tabel buku”.

1. Normalisasi Database
2. Dekomposisi mengubah ke 1NF

* Orders

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **OrderID** | **CustomerID** | **OrderDate** | **ShipDate** | **ShippingAddress** |
| 101 | 501 | 2023-01-10 | 2023-01-15 | 123 Main St |
| 102 | 502 | 2023-01-12 | 2023-01-17 | 456 Elm St |
| 103 | 503 | 2023-01-15 | 2023-01-20 | 789 Oak St |

* Order Details

|  |  |  |
| --- | --- | --- |
| **OrderID** | **ProductID** | **Quantity** |
| 101 | 201 | 2 |
| 102 | 202 | 1 |
| 103 | 203 | 3 |

1. Normalisasi dari 1NF ke 3NF

* 1NF

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **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 |

Permasalahan :

* Pengulangan -> Inkonsistensi

Proses lambat

* Penyimpangan :

S -> (Status, City) tapi tidak bisa insert data (S5, 30, JAKARTA) tanpa diikuti data P (khususnya) dan Q. Menghapus 1 baris data akan merusak keutuhan informasi.

* Solusi :

Dekomposisi menjadi : TPS1 dan TPS2

* 2NF
* 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 dapat menambah data (S5, 30, JAKARTA) dengan aman.
* Tapi masih ada penyimpangan :

Karena City -> Status maka 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 pengulangan 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
* Sebagian Pengulangan

Tidak berpotensi

* Lebih baik dari sebelumnya yang berlebihan

\*tidak menghilangkan semua yang berlebihan tapi meminimalisir

* 3NF

