



BASIS DATA

Teori dan Perancangan

Ni Luh Wiwik Sri Rahayu Ginantra • Ni Wayan Wardani
I Gusti Ayu Agung Mas Aristamy • I Wayan Dharma Suryawan
Dewa Putu Yudhi Ardiana • I Gede Iwan Sudipa
Ayu Manik Dirgayusari • Gede Surya Mahendra
Ni Kadek Ariasih • Wayan Gede Suka Parwita

BASIS DATA

Teori dan Perancangan

UU 28 tahun 2014 tentang Hak Cipta

Fungsi dan sifat hak cipta Pasal 4

Hak Cipta sebagaimana dimaksud dalam Pasal 3 huruf a merupakan hak eksklusif yang terdiri atas hak moral dan hak ekonomi.

Pembatasan Perfilman dan Pasal 26

Ketentuan sebagaimana dimaksud dalam Pasal 23, Pasal 24, dan Pasal 25 tidak berlaku terhadap:

- a. penggunaan kutipan singkat Ciptaan dan/atau produk Hak Terkait untuk pelaporan peristiwa aktual yang ditujukan hanya untuk keperluan penyediaan informasi aktual;
- b. Penggandaan Ciptaan dan/atau produk Hak Terkait hanya untuk kepentingan penelitian ilmu pengetahuan;
- c. Penggandaan Ciptaan dan/atau produk Hak Terkait hanya untuk keperluan pengajaran, kecuali pertunjukan dan Fonogram yang telah dilakukan Pengumuman sebagai bahan ajar; dan
- d. penggunaan untuk kepentingan pendidikan dan pengembangan ilmu pengetahuan yang memungkinkan suatu Ciptaan dan/atau produk Hak Terkait dapat digunakan tanpa izin Pelaku Pertunjukan, Produser Fonogram, atau Lembaga Penyiaran.

Sanksi Pelanggaran Pasal 113

1. Setiap Orang yang dengan tanpa hak dan/atau tanpa izin Pencipta atau pemegang Hak Cipta melakukan pelanggaran hak ekonomi Pencipta sebagaimana dimaksud dalam Pasal 9 ayat(1) huruf c, huruf d, huruf f, dan/atau huruf h untuk Penggunaan Secara Komersial dipidana dengan pidana penjara paling lama 3 (tiga) tahun dan/atau pidana denda paling banyak Rp500.000.000,00 (lima ratus juta rupiah).
2. Setiap Orang yang dengan tanpa hak dan/atau tanpa izin Pencipta atau pemegang Hak Cipta melakukan pelanggaran hak ekonomi Pencipta sebagaimana dimaksud dalam Pasal 9 ayat(1) huruf a, huruf b, huruf e, dan/atau huruf g untuk Penggunaan Secara Komersial dipidana dengan pidana penjara paling lama 4 (empat) tahun dan/atau pidana denda paling banyak Rp1.000.000.000,00 (satu miliar rupiah).

Basis Data

Teori dan Perancangan

Penulis:

Ni Luh Wiwik Sri Rahayu Ginantra, Ni Wayan Wardani
I Gusti Ayu Agung Mas Aristamy, I Wayan Dharma Suryawan
Dewa Putu Yudhi Ardiana, I Gede Iwan Sudipa
Ayu Manik Dirgayusari, Gede Surya Mahendra
Ni Kadek Ariasih, Wayan Gede Suka Parwita

Penerbit Yayasan Kita Menulis

Basis Data

Teori dan Perancangan

Copyright © Yayasan Kita Menulis, 2020

Penulis:

Ni Luh Wiwik Sri Rahayu Ginantra, Ni Wayan Wardani
I Gusti Ayu Agung Mas Aristamy, I Wayan Dharma Suryawan
Dewa Putu Yudhi Ardiana, I Gede Iwan Sudipa
Ayu Manik Dirgayusari, Gede Surya Mahendra
Ni Kadek Ariasih, Wayan Gede Suka Parwita

Editor: Janner Simarmata

Desain Sampul: Tim Kreatif Kita Menulis

Sampul: pexels.com

Penerbit

Yayasan Kita Menulis

Web: kitamenulis.id

e-mail: press@kitamenulis.id

WA: 0821-6453-7176

Ni Luh Wiwik Sri Rahayu Ginantra, dkk.

Basis Data: Teori dan Perancangan

Yayasan Kita Menulis, 2020

xiv; 156 hlm; 16 x 23 cm

ISBN: 978-623-6761-31-1

Cetakan 1, Oktober 2020

I. Basis Data: Teori dan Perancangan

II. Yayasan Kita Menulis

Katalog Dalam Terbitan

Hak cipta dilindungi undang-undang

Dilarang memperbanyak maupun mengedarkan buku tanpa

ijin tertulis dari penerbit maupun penulis

Kata Pengantar

Puji syukur kehadirat Tuhan Yang Maha Esa yang telah memberikan Rahmat dan HidayahNya, sehingga kami dapat menyelesaikan buku Basis Data : Teori dan Perancangan ini.

Basis data banyak diterapkan dan digunakan dalam merancang sebuah perangkat lunak. Dengan pemahaman tentang data, informasi, atribut dan membangun relasi antar atribut serta perancangan basis data yang baik maka diharapkan dapat menghasilkan perangkat lunak yang baik sehingga dapat mempermudah proses penyimpanan data serta pencarian data yang diperlukan oleh penggunanya.

Buku Basis Data ini terdiri dari 10 bab yaitu :

1. Pengenalan Basis Data
2. Data, Informasi, Hirarki Data dan Abstraksi Data
3. Ketergantungan Fungsionalitas dan Normalisasi
4. Model Data
5. Basis Data Relasional
6. Entitas, Atribut dan Relasi
7. Entity Relationship Diagram
8. Arsitektur Pemodelan Data
9. Perancangan Basis Data Perpustakaan
10. Perancangan Basis Data Penjualan

Akhir kata penulis menyampaikan terima kasih kepada teman-teman sejawat atas motivasi dan masukan yang bersifat membangun selama penulisan buku ini.

Denpasar Oktober 2020
Penulis

Daftar Isi

Kata Pengantar	v
Daftar Isi	vii
Daftar Gambar	xi
Daftar Tabel	xv

Bab 1 Pengenalan Basis Data

1.1 Definisi Basis Data	1
1.2 Sejarah Pemrosesan Basis Data (Database)	5
1.3 Istilah dalam Basis Data	6
1.4 Komponen Basis Data	8

Bab 2 Data, Informasi, Hirarki Data dan Abstraksi Data

2.1 Data	11
2.1.1 Data pada Basis data dan hubungannya	12
2.1.2 Jenis Data	12
2.2 Informasi	15
2.2.1 Kualitas Informasi	18
2.2.2 Konsep Dasar Informasi	18
2.3 Hirarki Data	19
2.4 Abstraksi Data	21

Bab 3 Ketergantungan Fungsional dan Normalisasi

3.1 Pendahuluan	23
3.2 Dependensi	24
3.2.1 Dependensi Fungsional	25
3.2.2 Dependensi Sepenuhnya	26
3.2.3 Dependensi Parsial	27
3.2.4 Dependensi Total	28
3.2.5 Dependensi Transitif	29
3.2.6 Diagram Dependensi Fungsional	30
3.3 Normalisasi	30

3.3.1 Bentuk Normal Pertama (1NF)	31
3.3.2 Bentuk Normal Kedua (2NF)	33
3.3.3 Bentuk Normal Ketiga (3NF)	35
3.3.4 Bentuk Normal Boyce-Codd (BCNF)	37
3.3.5 Bentuk Normal Keempat (4NF)	39
3.3.6 Bentuk Normal Kelima (5NF)	41

Bab 4 Model Data

4.1 Pengertian Model Data	43
4.2 Arti Penting Model Data	45
4.3 Evolusi Model Data	46
4.3.1 Model Hierarki dan Network	47
4.3.2 Model Relasional (Relational)	49
4.3.3 Model Entity-Relationship	53
4.3.4 Model Berorientasi Objek (Object-Oriented)	57
4.3.5 Object/ Relational	58

Bab 5 Basis Data Relasional

5.1 Pendahuluan	59
5.2 Definisi Basis Data Relasional	60
5.3 Struktur Basis Data Relasional	60
5.3.1 Struktur Tabel Basis Data Relasional	61
5.3.2 Kunci	63
5.3.3 Aturan Integritas	65
5.4 Hubungan Dalam Basis Data Relasional	67

Bab 6 Entitas, Atribut dan Relasi

6.1 Entitas	71
6.1.1 Isian Entitas	73
6.1.2 Himpunan Entitas	74
6.1.3 Jenis Entitas	74
6.2 Atribut	75
6.2.1 Pengelompokan Atribut	77
6.3 Relasi Antar Entitas	79
6.3.1 Simbol Relasi Antar Entitas	80
6.3.2 Relasi Berganda	83
6.3.3 Relasi Rekursif	84
6.3.4 Relasi Asosiatif	84

Bab 7 Entity Relationship Diagram

7.1 Pendahuluan.....	85
7.2 Konsep Entity Relationship Diagram	86
7.3 Tahapan pembuatan ERD.....	91
7.4 Contoh Kasus.....	91

Bab 8 Arsitektur Pemodelan Data

8.1 Pemodelan data dan Model Data.....	97
8.2 Tujuan Utama dari Pemodelan Data	99
8.3 Tingkatan Pemodelan Data	99
8.3.1 Conceptual Data Model	101
8.3.2 Logical Data Model.....	104
8.3.3 Physical Data Model	106
8.4 Alur Kerja Pemodelan Data	108
8.5 Perbandingan Conceptual, Logical dan Physical Data Model terhadap E-R Model	110
8.6 Keunggulan dan Kelemahan dari Model Data.....	110

Bab 9 Perancangan Basis Data Perpustakaan

9.1 Pengenalan Basis Data Perpustakaan.....	113
9.2 Tujuan dan Manfaat Perancangan Basis Data Perpustakaan.	114
9.3 Tahapan dalam perancangan basis data perpustakaan	115
9.3.1 Mengidentifikasi Entitas	115
9.3.2 Mengidentifikasi Jenis Relasi	115
9.3.3 Menentukan Kardinalitas.....	116
9.3.4 Mengidentifikasi Atribut dan Penentuan Primary Key	119
9.3.5 Membuat ERD (Entity Relationship Diagram).....	120
9.4 Tahapan Merubah ERD menjadi Struktur Tabel	121
9.4.1 Tabel Petugas.....	121
9.4.2 Tabel Anggota	121
9.4.3 Tabel Buku.....	122
9.4.4 Tabel RakBuku.....	123
9.4.5 Tabel Peminjaman.....	123
9.4.6 Tabel Pengembalian	124

Bab 10 Perancangan Basis Data Sistem Penjualan

10.1 Analisa Kebutuhan Sistem	125
10.1.1 Analisa Informasi.....	126
10.1.2 Analisa Fitur.....	127

10.2 Analisa Kebutuhan Tipe Data	128
10.3 Rancangan Basis Data	129
10.3.1 Notasi Crow's Foot	129
10.3.2 Conceptual Data Model	131
10.3.3 Logical Data Model.....	132
10.3.4 Physical Data Model	134
10.4 Implementasi Rancangan Basis Data	135
10.4.1 Implementasi pada MariaDB.....	135
 Daftar Pustaka.....	145
Biodata Penulis	151

Daftar Gambar

Gambar 1.1: Perbedaan lemari arsip dan Basis data	3
Gambar 1.2: Contoh Database Management System (DBMS)	4
Gambar 1.3: Model basis data (database) hirarki.....	6
Gambar 1.4: Komponen Basisdata.....	8
Gambar 2.1: Mengambil data langsung dari objek	13
Gambar 2.2: Data dari laporan keuangan	13
Gambar 2.3: Transformasi data hingga pengambilan keputusan.....	16
Gambar 2.4: Informasi dalam bentuk grafik.....	17
Gambar 2.5: Transformasi data menjadi informasi.....	17
Gambar 2.6: Hirarki data.....	19
Gambar 2.7: Contoh dari hirarki data.....	20
Gambar 2.8: Hubungan Level Abstraksi Data	21
Gambar 2.9: Contoh Abstraksi Data	22
Gambar 3.1: Contoh Penggambaran Relasi Dependensi Fungsional.....	30
Gambar 3.2: Langkah-langkah dalam Normalisasi	31
Gambar 4.1: Deskripsi dengan mengelompokkan data per mahasiswa.....	44
Gambar 4.2: Perkembangan model data	47
Gambar 4.3: Pemodelan data dalam bentuk tree	48
Gambar 4.4: Ilustrasi Model Data Hierarkhi menggunakan Basis Data Relasional.....	49
Gambar 4.5: Pemodelan data dalam bentuk network	49
Gambar 4.6: Contoh diagram skema	52
Gambar 4.7: Penggambaran himpunan entitas Mahasiswa.....	54
Gambar 4.8: Relasi yang menggambarkan hubungan antara Mahasiswa dengan Dosen	54
Gambar 4.9: Penggambaran atribut dari himpunan entitas Mahasiswa	55
Gambar 4.10: Penggambaran atribut komposit alamat yang disusun dari nama jalan, kota, dan provinsi	55
Gambar 4.11: Penggambaran atribut komposit alamat yang disusun dari nama jalan, kota, dan provinsi	56
Gambar 4.12: Penggambaran atribut umur yang nilainya dapat diketahui dari atribut tanggal_lahir.....	56

Gambar 5.1: Contoh Tabel atau Relation	61
Gambar 5.2: Contoh Tuple	61
Gambar 5.3: Contoh Attribute	62
Gambar 5.4: Degree dan Cardinality.....	62
Gambar 5.5: Relasi Antar Tabel	62
Gambar 5.6: Pelanggaran Aturan Integritas Entitas, terdapat nilai primary key yang sama	65
Gambar 5.7: Pelanggaran Aturan Integritas Entitas, terdapat nilai primary key yang null	65
Gambar 5.8: Pelanggaran Aturan Integritas Referensial, terdapat nilai yang tidak sesuai.....	67
Gambar 5.9: Hubungan one to one	68
Gambar 5.10: Hubungan one to many	68
Gambar 5.11: Hubungan Many to Many.....	69
Gambar 6.1 Contoh-contoh entitas berupa orang.....	72
Gambar 6.2 Contoh-contoh entitas berupa benda	73
Gambar 6.3 Contoh-contoh entitas berupa hal	73
Gambar 6.4. Entitas Barang	74
Gambar 6.5. Entitas Barang_Masuk	74
Gambar 6.6 Entitas super type dan sub type.....	75
Gambar 6.7 Contoh atribut pada entitas Barang	77
Gambar 6.8: Simbol kerelasian antar entitas (pilihan 1).....	81
Gambar 6.9: Simbol kerelasian antar entitas (pilihan 2).....	81
Gambar 6.10: Simbol kerelasian antar entitas (pilihan 3).....	82
Gambar 6.11: Contoh kelerasian antara entitas Karyawan dan Penjualan (pilihan 1).....	82
Gambar 6.12: Contoh kelerasian antara entitas Karyawan dan Penjualan (pilihan 2)	83
Gambar 6.13: Contoh kelerasian antara entitas Karyawan dan Penjualan (pilihan 3)	83
Gambar 6.14 Contoh Relasi Berganda.....	83
Gambar 6.15 Contoh kerelasian rekursif	84
Gambar 6.16 Contoh kerelasian Asosiasif.....	84
Gambar 7.1: Simbol Entiti set.....	86
Gambar 7.2: Simbol Relasi set	87
Gambar 7.3: Simbol atribut.....	87
Gambar 7.4: Relasi one to one.....	89
Gambar 7.5: Relasi one to many	90
Gambar 7.6: Relasi many to one	90

Gambar 7.7: Relasi many to many	90
Gambar 7.9: ERD sederhana	93
Gambar 7.10: ERD dengan kardinalitas	93
Gambar 7.11: ERD dengan berdasarkan kunci	94
Gambar 7.12: ERD dengan atribut.....	96
Gambar 8.1: Data Modeling	100
Gambar 8.2: Contoh Sederhana dari Conceptual Data Model pada Database Penjualan	103
Gambar 8.3: Contoh dari Conceptual Data Model pada Database Upload Video	103
Gambar 8.4: Contoh Sederhana dari Logical Data Model pada Database Penjualan	105
Gambar 8.5: Contoh dari Logical Data Model pada Database Upload Video	108
Gambar 8.6: Contoh Sederhana dari Physical Data Model pada Database Penjualan	107
Gambar 8.7: Contoh dari Physical Data Model pada Database Upload Video ...	108
Physical Data Model terhadap E-R Model.....	110
Gambar 9.1: Diagram Awal terhubung secara Entitas	116
Gambar 9.2: Kardinalitas relasi meminjam buku.....	117
Gambar 9.3: Kardinalitas relasi meminjam buku membentuk entitas komposit.....	117
Gambar 9.4: Kardinalitas relasi mengembalikanbuku.....	117
Gambar 9.5: Kardinalitas relasi mengembalikan buku membentuk entitas komposit.....	117
Gambar 9.6: Kardinalitas pada relasi memiliki rakbuku	117
Gambar 9.7: Kardinalitas pada relasi mengelola buku	118
Gambar 9.8: Kardinalitas pada relasi mengelola anggota	118
Gambar 9.10: ERD yang telah ditentukan kardinalitasnya	118
Gambar 9.11: ERD lengkap studi kasus Perpustakaan Sekolah	120
Gambar 10.1: Visualisasi Conceptual Data Model.....	131
Gambar 10.2: Visualisasi Logical Data Model	133
Gambar 10.3: Visualisasi Physical Data Model	134
Gambar 10.4: Visualisasi struktur table	143

Daftar Tabel

Tabel 1.1: Sejarah perkembangan pemrosesan basis data(database).....	5
Tabel 2.1: Data Mahasiswa peserta Mata Kuliah Basis Data.....	16
Tabel 3.1: Contoh Tabel yang Belum Ternormalisasi.....	24
Tabel 3.2: Contoh Tabel yang memiliki Dependensi Fungsional.....	26
Tabel 3.3: Contoh Tabel yang memiliki Dependensi Sepenuhnya.....	27
Tabel 3.4: Contoh Tabel yang memiliki Dependensi Parsial	28
Tabel 3.5: Contoh Tabel yang memiliki Dependensi Total	28
Tabel 3.6: Contoh Tabel yang memiliki Dependensi Transitif.....	29
Tabel 3.7: Contoh Tabel yang Belum Memenuhi 1NF	32
Tabel 3.8: Contoh Tabel yang Telah Memenuhi 1NF	32
Tabel 3.9: Contoh Tabel yang Masih Memiliki Redudansi	33
Tabel 3.10: Contoh Tabel yang Telah Memenuhi 1NF.....	33
Tabel 3.11: Contoh Tabel yang Belum Memenuhi 2NF	34
Tabel 3.12: Tabel Pembayaran Matkul 1 (Memenuhi 2NF).....	35
Tabel 3.13: Tabel Pembayaran Matkul 2 (Memenuhi 2NF).....	35
Tabel 3.14: Tabel Order Barang (Memenuhi 2NF)	36
Tabel 3.15: Tabel Oder (Memenuhi 3NF).....	37
Tabel 3.16: Tabel Item (Memenuhi 3NF).....	37
Tabel 3.17: Tabel Pemilihan MK (Memenuhi 1NF)	38
Tabel 3.18: Student Table (Memenuhi BCNF).....	38
Tabel 3.19: Professor Table (Memenuhi BCNF).....	39
Tabel 3.20: Tabel Matakuliah (Belum Memenuhi 4NF).....	39
Tabel 3.21: Tabel Matakuliah (Belum Memenuhi 4NF).....	40
Tabel 3.22: Tabel Subject (Memenuhi 4NF).....	40
Tabel 3.22: Tabel Alamat (Memenuhi 4NF).....	40
Tabel 4.1: Deskripsi menggunakan table.....	45
Tabel 4.2: Tabel dosen	50
Tabel 4.3: Tabel mahasiswa.....	51
Tabel 6.1: Contoh-Contoh Atribut	76
Tabel 7.1: Tabel Pegawai.....	87
Tabel 7.2: Tabel Keahlian.....	88

Tabel 7.3: Tabel penentuan relasi.....	92
Tabel 8.1: Perbandingan Conceptual Data Model, Logical Data Model dan	
Tabel 9.1: Tabel Mengidentifikasi Relasi Sistem Perpustakaan Sekolah	116
Tabel 9.2: Tabel identifikasi atribut dan penentuan primary key.....	119
Tabel 9.3: Tabel Petugas	121
Tabel 9.3: Tabel Anggota.....	122
Tabel 9.5: Tabel Buku.....	122
Tabel 9.6: Tabel RakBuku	123
Tabel 9.7: Tabel Buku.....	123
Tabel 9.8: Tabel Buku.....	124
Tabel 10.1: Simbol notasi Crow's Foot	130

Bab 1

Pengenalan Basis Data

1.1 Definisi Basis Data

Perkembangan teknologi 4.0 yang berkembang dengan pesat juga diimbangi dengan perkembangan perangkat lunak dan pemanfaatannya di berbagai bidang. Perkembangan pemanfaatan perangkat lunak dalam pengolahan data juga menjadi pendukung dalam kualitas perangkat lunak. Pengelolaan data yang baik akan memberikan informasi yang baik dalam mendukung keputusan bagi pihak terkait. Apa itu data dan informasi? Data adalah fakta, kumpulan angka, kata yang belum memiliki kegunaan, sedangkan informasi adalah kumpulan data yang telah diolah dan dapat dipergunakan dalam menunjang proses pengambilan keputusan bagi penggunanya.

Pada dasarnya berbicara tentang pengolahan data secara digital dalam keterkaitan perangkat lunak kita mengenal istilah basis data (database). Tidak satupun dari perkembangan baru ini yang menghilangkan kebutuhan akan aplikasi pengolahan database klasik yang paling vital bagi kepentingan bisnis dan lainnya. Basis data (database) secara umum adalah item data yang saling berhubungan satu dengan yang lainnya yang diorganisasikan berdasarkan skema atau struktur tertentu yang dapat disimpan dalam perangkat keras (hardware) dan dengan menggunakan perangkat lunak dalam manipulasi untuk kegunaan tertentu (Irmansyah, 2003). Basis data (database) juga dapat diartikan kumpulan records yang saling berhubungan yang disimpan secara bersama sedemikian rupa tanpa pengulangan (redundancy) yang tidak perlu untuk memenuhi kebutuhan (Kroenke, 2003). Pengertian ini digambarkan seperti lemari arsip yang menyimpan beberapa arsip disebuah ruangan yang

dikelela secara konvensional sedangkan basis data penyimpanan data secara digital yang diproses dengan menggunakan *Database Management System* (DBMS).

Pengolahan data berbasis berkas memiliki kelemahan yaitu :(Kadir.A, 2008)

1. Data terpisah-pisah

Data pada sistem pemrosesan yang berbasis berkas terkadang terpisah-pisah (tersebar) pada berkas yang lain yang ada. Kemungkinan untuk proses pencarian berkas akan menjadi sulit karena posisi yang tersebar pada perangkat yang ada.

2. Adanya duplikasi data (redundancy)

Karena penyebaran data berbasis berkas menjadikan data akan tersebar sehingga ada kemungkinan terciptanya data duplikasi. Tanpa adanya koordinasi akan membuat tercipta duplikasi data. Duplikasi data yang berlebihan akan membuat konsekuensi yang tidak baik.

3. Terjadinya ketidakkompatibelan berkas.

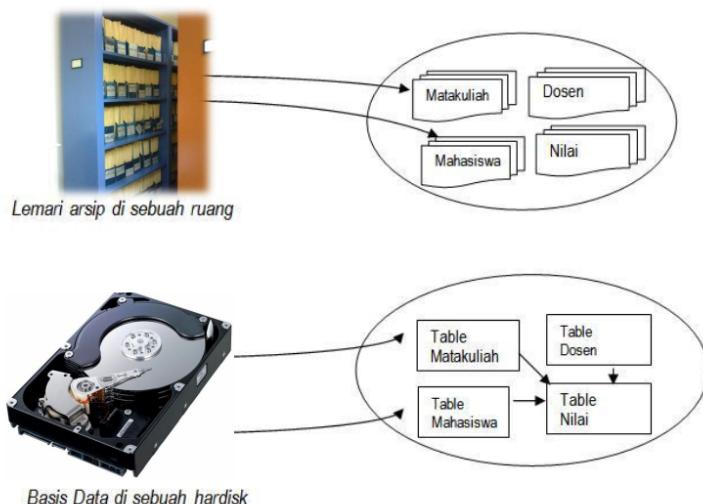
Banyaknya bahasa pemrograman yang dipergunakan dalam mengakses data pada sistem yang berbasis berkas akan membuat perbedaan format. Perbedaan format akan menimbulkan permasalahan dalam proses mendapatkan informasi yang ingin didapatkan dan dapat menyita waktu.

4. Ketergantungan terhadap pemrograman.

Proses pembuatan laporan pada sistem berbasis berkas akan sangat tergantung pada pemrograman. Hal ini tentunya akan berbeda ketika menggunakan database.

5. Membutuhkan space/ penyimpanan yang lebih.

Karena adanya kemungkinan duplikasi data dan penyebaran data akan membuat sistem berbasis berkas membutuhkan space penyimpanan yang lebih daripada menggunakan database.



Gambar 1.1: Perbedaan lemari arsip dan Basis data

Mengapa menggunakan basis data (database)? Ada beberapa alasan mengapa basis data dipergunakan.

1. Untuk mengurangi terjadinya duplikasi (redundancy) data.
2. Komponen dasar dalam sistem informasi sehingga dapat menyediakan informasi yang diperlukan.
3. Kecepatan dan kemudahan dalam proses penyimpanan data, perubahan data serta pencarian
4. Mengurangi tempat penyimpanan (space)
5. Keamanan (security)

Data yang bersifat rahasia tidak disalah gunakan oleh orang yang tidak memiliki hak akses. Pada database dapat diberikan hak akses kepada siapa saja dapat mengakses database tersebut.

6. Keakuratan (accuracy)

Konsistensi data yang dalam relasi dan type data akan memudahkan dalam proses yang ingin dilakukan.

Disisi lain penggunaan basis data memiliki kekurangan. Adapun kekurangan dari penggunaan basis data adalah:

1. Biaya mahal

Perancangan dan implementasi penggunaan database tentunya membutuhkan biaya. Dari sisi perangkat keras ketersediaan perangkat seperti CPU yang memadai tentunya membutuhkan biaya yang tidak bisa dikatakan murah. Sedangkan dari sisi perancangan membutuhkan analist sistem yang juga dibayar.

2. Maintenance back up yang membutuhkan keahlian khusus.

Menjaga data dari kerusakan dan kehilangan perlu dilakukan perawatan secara berkala, yang tidak sembarangan bisa dilakukan oleh orang. Dibutuhkan orang yang memahami dan paham tentang database untuk melakukan maintenance.

Lalu apa itu *Database Management System* (DBMS)? *Database Management System* (DBMS) adalah sebuah perangkat lunak yang dirancang untuk melakukan pemrosesan suatu basis data dan menjalankan operasi yang diminta oleh pengguna. Contoh DBMS yakni Oracle dari Oracle Corporation, Microsoft Access serta SQL Server dari Microsoft, MySQL dan lainnya. (Kroenke, 2003). DBMS ini umumnya bersifat komersil, namun ada DBMS yang tidak bersifat komersil seperti MySQL atau PostgreSQL. Fasilitas DBMS non komersil tidak selengkap DBMS komersil (Kadir.A, 2008). Sehingga basis data (database) tidaklah sama dengan *Database Management System* (DBMS).



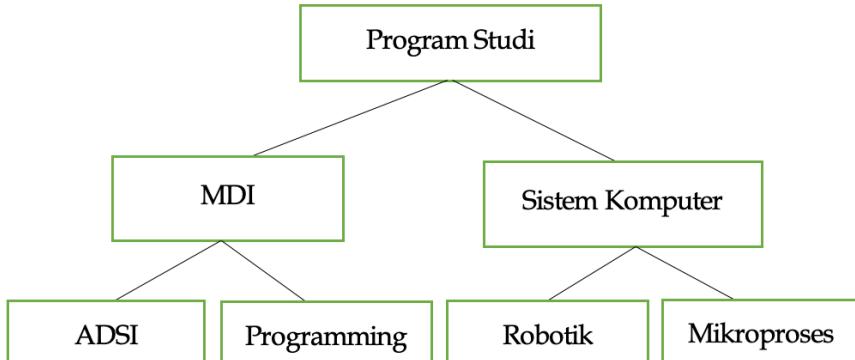
Gambar 1.2: Contoh Database Management System (DBMS)

1.2 Sejarah Pemrosesan Basis Data (Database)

Pertengahan tahun 1960an hampir semua data komputer pada masa ini menggunakan pita magnetik, namun karena pita magnetik hanya dapat diakses hanya secara berurutan maka data disimpan kedalam bentuk daftar. Perkembangan basis data(database) dapat dilihat pada tabel 1.

Tabel 1.1: Sejarah perkembangan pemrosesan basis data(database) (Kroenke, 2003)

Tahun	Teknologi	Perkembangan
Sebelum 1968	Pemrosesan File	Pemrosesan sebelum menggunakan database. Karakteristiknya masih menggunakan pita magnetik
1968-1980	Hierarkis dan model network	Era pemrosesan <i>database</i> non-relasional. Model data network yang terkemuka adalah model CODASYL DBTG dan DBMS yang popular adalah DBMS network
1980 -sekarang	Model data relasional	1970 pertama kali model data relasional dipublikasi dan tahun 1980 mulai dikomersialkan. SQL menjadi Bahasa relasional standar.
1982	Produk DBMS mikrokomputer pertama	Ashton-tate mengembangkan produk dBBase; Borland membuat Paradox
1991	Microsoft ship Access	Produk DBMS personal dari Microsoft.
1995	Aplikasi pertama database internet	Database menjadi komponen kunci dari aplikasi internet.
1997	Penerapan XML pada pemrosesan database	XML digunakan untuk memecahkan permasalahan database jangka panjang.



Gambar 1.3: Model basis data (database) hirarki.

1.3 Istilah dalam Basis Data

Dalam basis data ada beberapa istilah dasar yang perlu diketahui.

1. **Entitas**, biasanya berupa objek yang berwujud, seperti: orang, rumah, mobil dan lainnya. Entitas terbagi menjadi dua yaitu entitas kuat dan lemah.
2. **Atribut** adalah karakteristik yang melekat yang menjelaskan secara detail entitas.

Contoh :

Entitas Sepeda Motor = (**no_rangka, warna, type**)

Entitas pegawai = (**no_pegawai, nama, alamat, tgl_lahir**)

3. **Record** adalah isian data yang saling berhubung.

Contoh :

Entitas Sepeda motor dengan atribut no_rangka, model, warna dan type akan memiliki record "K1223CO, hitam, matic"

4. **File** merupakan kumpulan dari record yang menggambarkan entitas.

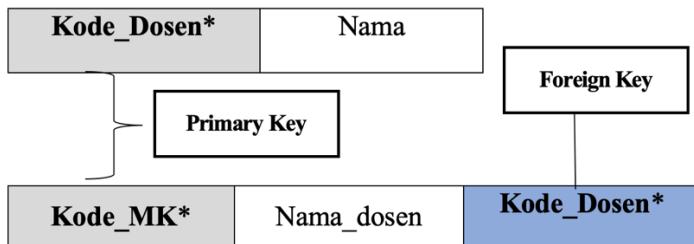
5. **Key** bagian dari record yang dipergunakan untuk proses akses dan menemukan record yang dimaksud. Ada beberapa jenis key dalam database yaitu :
- Primary Key** umumnya bersifat unik dan field yang mengidentifikasikan sebuah record.

Contoh Primary Key

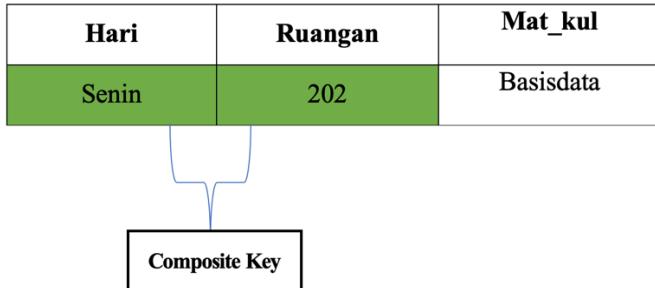
Primary Key		
Kode_barang*	Nama_barang	Jumlah_barang

- Foreign Key** (Kunci Tamu) adalah field yang bukan kunci dan biasanya disebut sebagai kunci tamu yang merupakan kunci pada file yang lain.

Contoh Foreign Key

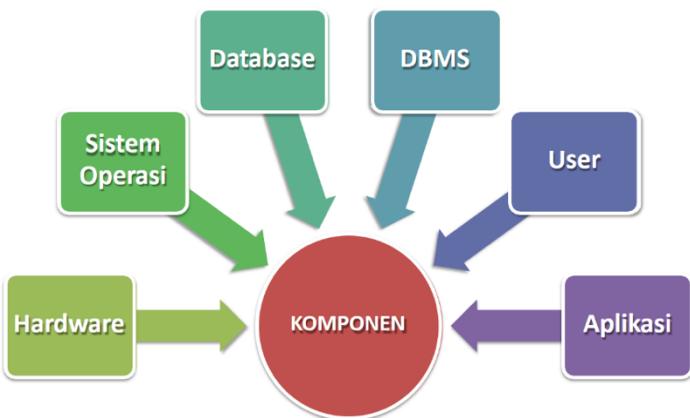


- Candidate Key** merupakan atribut yang mengidentifikasikan secara unik sebuah entitas. Syarat sebuah candidate key adalah: unique identifier dan no redundancy.
- Composite Key** adalah sebuah kunci yang terdiri dari 2 atribut atau lebih yang mengidentifikasikan suatu entitas. Contoh dari composite key adalah pada field hari dan ruangan pada tabel matakuliah.



1.4 Komponen Basis Data

Dalam sistem basidata terdapat komponen-komponen pendukung yang dapat mendukung proses sistem basis data bekerja. Komponen tersebut di antaranya: Hardware/perangkat keras, Sistem Operasi, Database, *Database Management System* (DBMS), User, dan Aplikasi.



Gambar 1.4: Komponen Basisdata

1. Perangkat keras/hardware

Perangkat keras/hardware merupakan komponen yang mendukung proses basis data. Perangkat keras seperti memory, hardisk, CPU menjadi penunjang dalam kecepatan untuk proses basis data.

2. Sistem Operasi (SO)

Merupakan program untuk menjalankan aplikasi pada sebuah komputer. Sistem operasi ini berfungsi menunjang aplikasi DBMS yang akan dipergunakan. DBMS dapat berjalan dengan baik pada sistem operasi yang sesuai. Contoh dari sistem operasi ini adalah: Windows, Linux, Mac OS dan lainnya yang dikembangkan oleh perusahaan teknologi dunia seperti Apple.

3. Database

Kumpulan dari data/field yang saling terkait yang dapat dimanipulasi dan disimpan. Setiap database akan memiliki indeks, tabel dan sejumlah objek. Struktur tabel yang baik akan menunjang database mudah dikembangkan dan dijalankan.

4. Database Management System (DBMS)

Database Management System (DBMS) merupakan aplikasi yang dipergunakan untuk mengelola database. Contoh dari aplikasi DBMS adalah MySQL, SQLServer, Microsoft Access, Oracle dan lainnya. Pemilihan serta penggunaan DBMS yang sesuai dengan kebutuhan database yang akan dirancang juga perlu dipertimbangkan.

5. User

Merupakan pengguna (user) yang akan menggunakan database. User dapat dibagi menjadi beberapa jenis yaitu: database administrator, database designer, end-user dan *software engineering*. *Database administrator* (DBA) adalah seseorang yang akan mengola database tersebut seperti melakukan analisa sistem dan menjamin integritas data. Database designer adalah seseorang yang merancang database sedangkan *software engineering* adalah seorang yang mengimplementasikan database ke dalam sebuah sistem.

End-user juga dapat dibagi menjadi beberapa jenis yaitu :

- a. ***User mahir/ casual end user*** adalah user yang berinteraksi dengan sistem dan menyatakan dengan query dengan bahasa query yang telah disediakan oleh DBMS.

- b. *Naive /Parametric end user* adalah user yang mengerjakan terkait update data seperti pada bank teller, bagian reservasi dan lain sebagainya.
 - c. *Sophisticated end user* umumnya adalah user yang melengkapi kebutuhan database seperti data analyst, engineer, bisnis analyst dan sebagainya.
 - d. *Stand alone user* adalah pengguna yang mengelola databasenya sendiri dengan menggunakan perangkat lunak yang telah tersedia dan tersedia antarmuka sehingga mempermudah dalam menggunakan database tersebut (Geeksforgeeks, 2019).
6. Aplikasi

Database biasanya dipergunakan untuk membantu mempermudah proses pada program aplikasi. Seperti pada proses pencarian, penyusunan laporan transaksi penjualan dan pembelian dan aplikasi lainnya. Saat ini hampir aplikasi yang umumnya dipergunakan seperti pada perusahaan retail hotel, perbankkan menggunakan database sebagai media penyimpanan datanya.

Bab 2

Data, Informasi, Hirarki Data dan Abstraksi Data

2.1 Data

Data merupakan salah satu komponen penting sistem basis data selain hardware, software, dan user. Data menyatakan fakta – fakta yang mewakili kejadian, aktivitas, transaksi dan juga deskripsi mengenai suatu objek/benda yang terekam dan tersimpan dalam media komputer. Data dinyatakan dengan nilai angka, nilai deretan, nilai karakter, atau nilai simbol. Data tidak hanya dalam bentuk teks tetapi juga dalam bentuk dokumen, gambar, suara, ataupun video.

Berikut ini adalah ciri – ciri data dalam sistem basis data yaitu:

1. Data disimpan secara terintegrasi.

Kumpulan dari berbagai macam berkas dari aplikasi berbeda dan disusun sedemikian rupa, di mana bagian yang rangkap (redundant) akan dihilangkan.

2. Data dapat digunakan secara bersama – sama (shared).

Untuk aplikasi yang berbeda, masing-masing bagian dari basis data dapat diakses oleh user dalam waktu yang bersamaan.

2.1.1 Data pada Basis data dan hubungannya

Terdapat 3 jenis data pada sistem basis data yaitu :

1. Data operasional dari suatu organisasi, berupa data yang disimpan di dalam basis data.
2. Data masukan, data dari luar sistem yang dimasukkan melalui peralatan masukan seperti keyboard yang dapat mengubah data operasional.
3. Data keluaran, keluaran yang berupa laporan melalui peralatan keluaran sebagai hasil dari dalam sistem yang mengakses data operasional.

2.1.2 Jenis Data

a. Menurut Cara Memperolehnya

1. Data Primer

Data primer merupakan data yang diambil secara langsung dari objek, di mana sumber data yang langsung memberikan data kepada pengumpul data tanpa melalui media perantara melalui proses wawancara, menjawab kuesioner ataupun forum diskusi.

Bentuk data primer berupa hasil kuesioner, hasil survei, hasil pengujian, hasil observasi dan opini subyek secara individual atau kelompok.

Contoh :

- Melakukan wawancara secara langsung terhadap para pedagang pasar tradisional untuk meneliti daya beli konsumen selama pandemic.
- Observasi dan pengamatan proses antrian layanan di kantor catatan sipil.



Gambar 2.1: Mengambil data langsung dari objek

2. Data Sekunder

Data sekunder merupakan data yang didapat secara tidak langsung dari sumber data atau objek penelitian di mana data sekunder merupakan data yang mendukung keperluan data primer seperti literature bacaan, buku, majalah, koran, artikel jurnal ataupun laporan – laporan.

Contoh

- Hasil riset dalam bentuk data statistic dari majalah ataupun surat kabar.
- Menggunakan data dari laporan keuangan.



Gambar 2.2: Data dari laporan keuangan

b. Berdasarkan Sumbernya

1. Data Internal

Data internal merupakan data yang mewakili situasi dan kondisi suatu organisasi di mana fakta dikumpulkan, dicatat, dan disimpan secara internal.

Contoh :

- Data keuangan
 - Data pegawai
 - Data produksi
2. Data Eksternal

Data eksternal merupakan data yang mewakili dan situasi serta kondisi yang berada di luar organisasi. Selanjutnya data eksternal terbagi menjadi dua kategori yaitu data primer dan data sekunder.

Contoh :

- Data jumlah pemakaian suatu produk oleh konsumen
 - Tingkat preferensi pelanggan
 - Nilai GDP dari Indonesia
- c. Jenis Data Berdasarkan Sifatnya

1. Data Kuantitatif

Data kuantitatif merupakan data dalam bentuk angka atau skor di mana data diperoleh dari penggunaan alat pengumpul data ataupun sekumpulan pertanyaan yang telah diberi bobot. Data kuantitatif memiliki kecenderungan dapat dianalisa dengan teknik statistik.

Contoh :

- Tinggi badan rata-rata perempuan Indonesia adalah 155 cm
2. Data Kualitatif

Data kualitatif adalah data yang disajikan dalam bentuk kata-kata ataupun gambar yang mengandung makna yang menjelaskan tentang suatu kualitas dari sebuah kejadian. Data kualitatif tidak dapat diukur secara numerik.

Contoh :

- Perempuan Amerika lebih tinggi dari perempuan Indonesia
- d. Berdasarkan Waktu Pengumpulannya
- 1. Data Berkala (Time Series)

Data berkala merupakan data yang dikumpulkan dan dicatat dari waktu ke waktu (berkala) , biasanya digunakan untuk melihat perkembangan dari waktu ke waktu.

Contoh :

- Jumlah keuntungan perusahaan dari tahun ke tahun.
- 2. Data Cross Section

Data cross section merupakan data yang didapat pada waktu yang telah ditentukan untuk mendapatkan gambaran keadaan atau kegiatan pada saat itu juga.

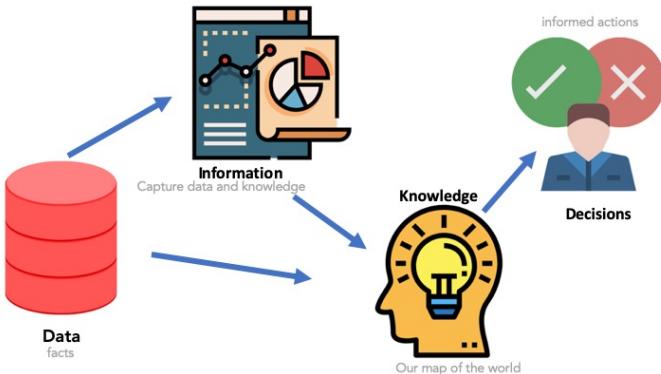
Contoh :

- Mengukur suhu tubuh saat ini pada populasi tertentu.

2.2 Informasi

Acapkali istilah data dan informasi menjadi rancu dan dipertukarkan dalam penyebutannya terutama dalam penggunaan data yang berjumlah kecil walaupun sebenarnya data dan informasi adalah sesuatu yang berbeda. Menurut Davis (1999), Informasi merupakan hasil dari mengolah data sehingga menjadi bentuk yang penting dan bermanfaat bagi penerimanya sebagai dasar dalam pengambilan keputusan yang berakibat secara langsung saat itu juga atau secara tidak langsung pada masa yang akan datang.

Dari pengertian di atas, bahwa informasi berasal dari data yang diolah kemudian bertransformasi menjadi bentuk yang bermanfaat bagi pengambilan keputusan.



Gambar 2.3: Transformasi data hingga pengambilan keputusan

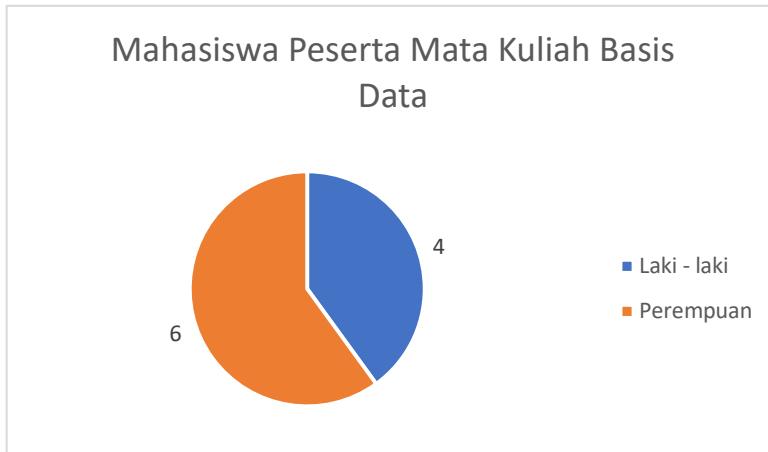
Berikut merupakan contoh sederhana data mahasiswa yang mengambil mata kuliah basis data.

Tabel 2.1: Data Mahasiswa peserta Mata Kuliah Basis Data

20101001	Dewi Maruti
20101002	Yoga Saputra
20101003	Salasika Tantri
20101004	Adira Pratiwi
20101005	Kenzia Tan
20101006	Surya Saputra
20101007	Widya Widana
20101008	Darma Putra
20101009	Kusuma Wardani
20101010	Ana Victoria

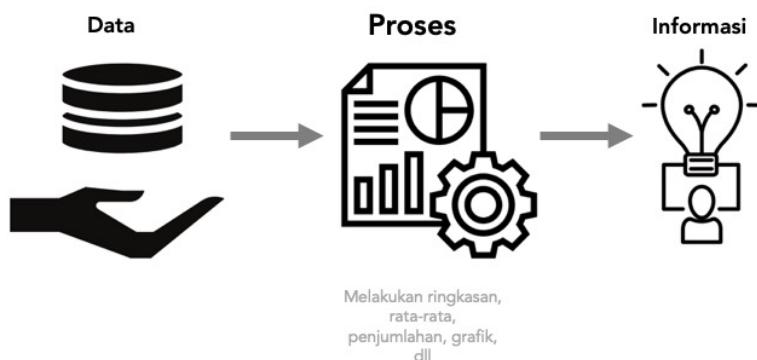
Dari tabel data di atas, jika data diolah maka kira – kira informasi apa yang bisa kita dapatkan ?

Contoh informasi yang bisa didapatkan dari data diatas adalah “Jumlah mahasiswa peserta mata kuliah basis data adalah 10 orang terdiri dari 4 orang mahasiswa laki-laki dan 6 orang mahasiswa perempuan”.



Gambar 2.4: Informasi dalam bentuk grafik

Informasi jumlah peserta mata kuliah basis data dapat berguna bagi penerima informasi yang relevan dan dapat dimanfaatkan untuk mengambil keputusan.



Gambar 2.5: Transformasi data menjadi informasi

2.2.1 Kualitas Informasi

Kualitas dari informasi dipengaruhi oleh 3 unsur (Sutabri, 2012), yaitu :

- 1. Akurat**

Informasi harus dijamin kebenarannya serta tidak menyesatkan bagi penerima informasi serta jelas mencerminkan maksudnya karena kemungkinan informasi akan mengalami kerusakan atau gangguan.

- 2. Tepat Waktu**

Informasi yang usang tidak memiliki nilai dan kualitas yang baik bagi pengambilan keputusan sehingga tidak berguna lagi pada saat pengiriman informasi dari sumber informasi ke penerima informasi.

- 3. Relevan**

Informasi yang diterima oleh penerima informasi harus sesuai, tepat sasaran dan bermanfaat.

2.2.2 Konsep Dasar Informasi

Konsep dasar informasi dikelompokkan menjadi 3 bagian, yaitu (Sutabri, 2012) :

- 1. Informasi Strategis**

Informasi ini diperlukan untuk pengambilan keputusan dalam waktu jangka panjang, yang mencakup informasi eksternal, rencana perluasan perusahaan, dan lain sebagainya.

- 2. Informasi Taktis**

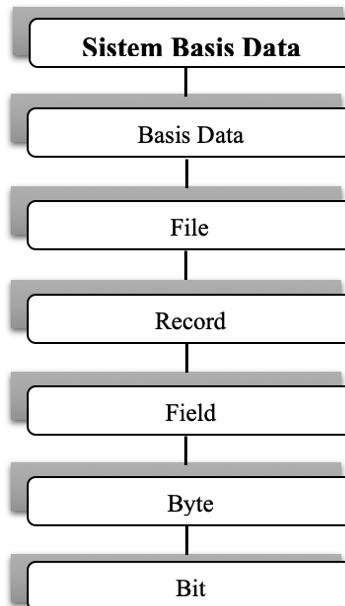
Informasi ini diperlukan untuk pengambilan keputusan dalam jangka waktu menengah.

- 3. Informasi Teknis**

Informasi ini diperlukan untuk kebutuhan operasional sehari-hari.

2.3 Hirarki Data

Pengorganisasian data berdasarkan tingkat kompleksitas nilai data dapat dibagi menjadi 6 tingkatan di mana disusun dalam sebuah hirarki data, sebagaimana gambar berikut :



Gambar 2.5: Hirarki data

1. Bit

Sistem biner yang terdiri atas 2 macam (0 dan 1), di mana sebagai dasar komunikasi antara komputer dan manusia.

2. Byte

Satuan penyimpanan data dalam komputer. (1 byte = 8 bit = 1 karakter).

3. Field/atribut/data item

Sekumpulan byte atau karakter yang mempunyai makna dan menjadi unit terkecil yang disebut data.

4. Record

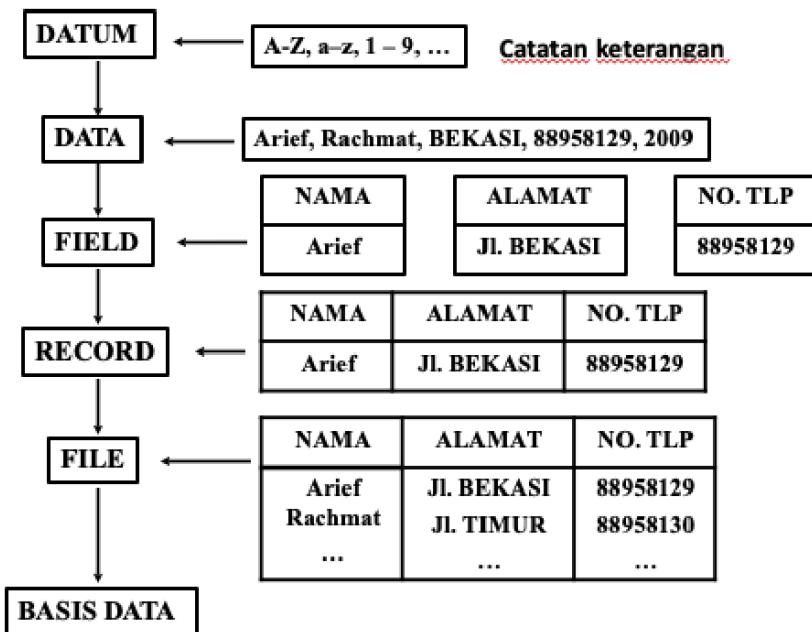
Merupakan sekumpulan field/atribut/data item yang saling berhubungan terhadap obyek tertentu dan dihitung dalam satuan baris.

5. Berkas/file

Merupakan sekumpulan record yang sama jenis secara relasi dan tersimpan dalam media penyimpanan sekunder.

6. Basis data

Merupakan sekumpulan dari bermacam-macam tipe record yang memiliki hubungan antar record dan rinci data terhadap obyek tertentu di mana tersimpan di dalam computer secara terorganisir.



Gambar 2.6: Contoh dari hirarki data

2.4 Abstraksi Data

Abstraksi data adalah tingkatan/level dalam melihat bagaimana menampilkan data dalam sebuah basis data.

Tingkatan abstraksi data oleh pengguna dikelompokkan menjadi 3 tingkatan sebagai berikut :

1. Tingkatan Fisik.

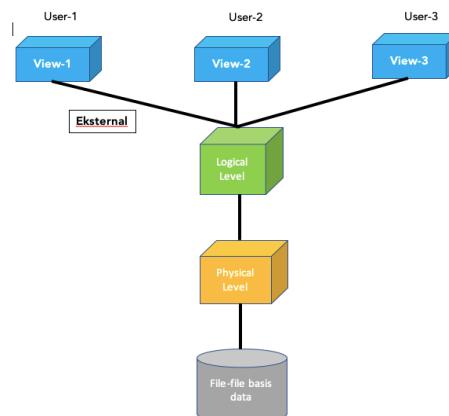
Tingkatan fisik adalah Level paling rendah dalam abstraksi data, yang menunjukkan bagaimana suatu data disimpan.

2. Tingkatan Logis/Konseptual.

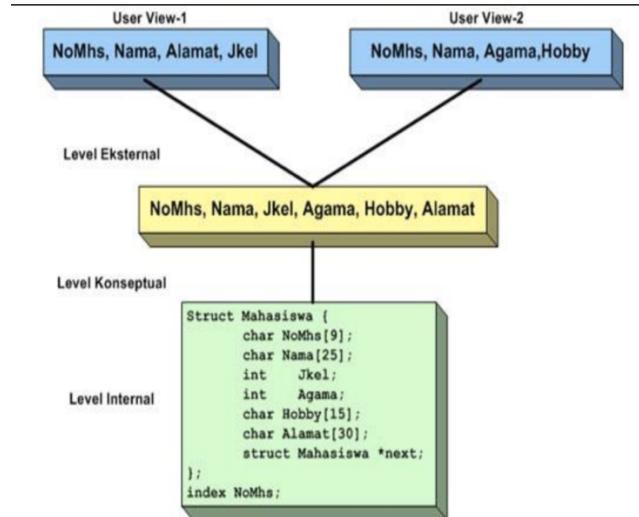
Tingkatan logis adalah tingkat abstraksi yang lebih tinggi berikutnya. Tingkatan ini menggambarkan hubungan antara suatu data dengan data lain dan data apa saja yang tersimpan di dalam basis data.

3. Tingkatan View.

Tingkatan ini adalah tingkatan abstraksi yang paling tinggi, dan yang paling sederhana di antara tingkatan lainnya. Tidak menampilkan data dari basis data secara keseluruhan tetapi hanya sebagian data dari basis data. User tidak membutuhkan semua data yang ada dalam basis data.



Gambar 2.7: Hubungan Level Abstraksi Data



Gambar 2.8: Contoh Abstraksi Data

Bab 3

Ketergantungan Fungsional dan Normalisasi

3.1 Pendahuluan

Merancang suatu database yang baik merupakan hal yang diharapkan oleh setiap developer sistem informasi. Rancangan database yang baik memiliki ciri-ciri yaitu dapat mencegah adanya anomali, tidak konsistennya data (inkonsistensi), memperkecil peluang munculnya duplikasi data atau tersimpannya data yang sama secara berulang (redundansi data), dan tidak banyak membutuhkan ruang penyimpanan. Untuk dapat membuat database dengan ciri-ciri seperti yang telah disebutkan, maka perlu dilakukannya normalisasi data. Normalisasi adalah proses pembentukan struktur basis data, sehingga sebagian besar ambiguity bisa dihilangkan (Puspitasari, Rahmad and Astiningrum, 2016). Normalisasi juga merupakan pada logical desain sebuah basis data relasional yang mengelompokkan atribut dari suatu tabel, sehingga membentuk struktur tabel yang normal. Tujuan dari dilakukannya normalisasi, tidak lain adalah untuk meminimalisir adanya redundansi data, atau tersimpannya data yang sama secara berulang. Pertanyaannya adalah, mengapa redundansi perlu untuk diminimalisir? Karena, redundansi dapat menimbulkan permasalahan, yaitu anomali. Anomali adalah masalah yang muncul dalam relasi pada saat terjadi operasi pembaharuan data dalam relasi (Connoly, Begg

and Strachan, 1996). Untuk lebih jelasnya, mari kita simak tabel 3.1. Apakah tabel berikut merupakan tabel yang berstruktur baik atau masih belum baik?

Tabel 3.1: Contoh Tabel yang Belum Ternormalisasi

NIM	Nama Mahasiswa	Semester	Iuran SPP
1204505019	Ani	1	Rp 2.000.000
1204505021	Ana	3	Rp 2.750.000
1204505034	Deva	3	Rp 2.750.000
1204505038	Devi	1	Rp 2.000.000
1204505045	Mirna	1	Rp 2.000.000

Jawabannya adalah, tabel x adalah tabel yang belum berstruktur baik. Mengapa? Karena, masih terdapat redundansi atau tersimpannya data yang sama secara berulang. Di manakah letak redundansinya? Ada di kolom Semester dan Iuran SPP. Mengapa data yang ada di kolom Semester dan Iuran SPP dikatakan redundan? Karena, di kolom Semester, sudah diketahui bahwa mahasiswa yang sedang menempuh kuliah di semester 1, harus membayar iuran spp sebesar Rp. 2000.000,-, namun jumlah iuran tersebut di sebutkan secara berulang. Permasalahannya ada pada saat akan melakukan update data pada tabel, ketika nominal jumlah iuran spp akan dinaikkan, maka data yang harus diupdate bukan hanya satu data, melainkan ada tiga data. Jika dalam satu tabel terdapat 1000 data, apakah memungkinkan untuk merubahnya secara manual? Ya memungkinkan, namun hal tersebut sangat tidak efektif. Maka dari itu, perlu dibuat tabel untuk memisahkan iuran spp dan dibuat tabel baru yang khusus berisikan data Semester beserta Iuran SPP.

3.2 Dependensi

Tidak jarang, banyak pemula yang mengalami kesulitan dalam melakukan normalisasi data. Apakah penyebabnya? Penyebab utama yang sering ditemukan adalah karena belum menguasai dan memahami dengan baik konsep dari ketergantungan (dependency). Dependensi adalah ketergantungan, ketergantungan dari suatu atribut ke atribut lainnya. Jika memiliki sebuah tabel, harus dapat dipastikan tabel tersebut memiliki ketergantungan dengan tabel lainnya. Ketergantungan erat kaitannya dengan normalisasi, karena

ketergantungan merupakan konsep yang mendasari proses normalisasi (Iyer, 2015).

Dependensi dibagi menjadi beberapa jenis, yaitu:

1. Dependensi fungsional
2. Dependensi sepenuhnya
3. Dependensi parsial
4. Dependensi total
5. Dependensi transitif

3.2.1 Dependensi Fungsional

Dependensi atau ketergantungan fungsional atau sering disebut dengan Functional Dependency, merupakan kondisi di mana satu atribut atau beberapa atribut dari suatu relasi yang keberadaannya bergantung pada atribut lain (Setiadi, 2017). Definisi lain menyebutkan, ketergantungan fungsional digunakan untuk menggambarkan atau mendeskripsikan bentuk normal atas suatu relasi. Hal ini dapat membantu developer sistem informasi dalam menjaga kualitas data dalam database. Selain itu, ketergantungan fungsional juga berperan penting untuk dapat membedakan mana desain database yang baik dan yang buruk. Ketergantungan fungsional diperoleh berdasarkan hubungan antar atribut data. Adapun beberapa kegunaan dari ketergantungan fungsional adalah untuk memeriksa apakah semua relasi telah sesuai dengan standar dari ketergantungan fungsional yang ditetapkan, menetapkan batasan gugus relasi yang berlaku, menentukan kunci relasi, dan untuk melakukan normalisasi atas suatu tabel relasional. Ketergantungan fungsional dilambangkan dengan notasi \rightarrow (Iyer, 2015).

Berikut merupakan contoh analogi dari ketergantungan fungsional:

Contoh:

1. Misalkan terdapat relasi A dengan dua buah atributnya berupa X dan Y, sehingga notasinya $A(X,Y)$.
2. Pada kondisi tersebut, Y dikatakan memiliki dependensi fungsional terhadap X apabila setiap nilai dalam X berhubungan dengan satu nilai yang sama pada Y. Sehingga, notasinya $X \rightarrow Y$.
3. Kondisi ini menandakan jika Y memiliki ketergantungan terhadap X. Untuk contoh lebih jelasnya, mari kita perhatikan tabel 3.2.1 berikut.

Tabel 3.2: Contoh Tabel yang memiliki Dependensi Fungsional

Matakuliah	NIM	Nama Mahasiswa	Nilai
Algoritma	09092020	Ani	A
Pemrograman	09092020	Ani	B
Pemrograman	09782020	Ana	A
Algoritma	09552020	Devi	B

Pada tabel 3.2 tersebut, ketergantungan fungsional terletak pada nilai dari kolom NIM → Nama Mahasiswa. Artinya adalah nilai dari satu NIM menentukan nilai dari satu Nama Mahasiswa, atau Nama Mahasiswa memiliki ketergantungan fungsional dengan NIM.

Dependensi fungsional tidak selalu memiliki ketergantungan pada satu atribut, artinya sebuah atribut bisa dibuat bergantung pada lebih dari satu atribut. Contohnya, pada tabel di atas berlaku notasi {Matakuliah, NIM} → Nilai, yang artinya satu nilai dari NIM dan Matakuliah, menentukan Nilai mahasiswa. Mengapa demikian? Padahal, secara definisi dikatakan Y memiliki dependensi fungsional terhadap X apabila setiap nilai dalam X berhubungan dengan satu nilai yang sama pada Y. Namun, pada tabel tersebut ditampilkan bahwa satu NIM, tidak hanya memiliki satu Nilai saja, melainkan dapat memiliki dua Nilai. Jika menggunakan Matakuliah saja, Matakuliah pun memiliki lebih dari satu Nilai. Lalu mengapa Matakuliah dan NIM menentukan Nilai mahasiswa? Karena, setelah diamati, satu mahasiswa tidak hanya memiliki satu Matakuliah dan Nilai yang mutlak, melainkan satu mahasiswa dapat memiliki Matakuliah dan Nilai yang lebih dari satu. Itu artinya, Nilai yang diperoleh mahasiswa tidak hanya ditentukan melalui Matakuliah saja, melainkan ditentukan oleh Matakuliah dan NIM.

3.2.2 Dependensi Sepenuhnya

Dependensi atau ketergantungan sepenuhnya ditandai dengan kondisi apabila suatu atribut memiliki dependensi fungsional terhadap atribut lainnya, namun tidak memiliki dependensi terhadap bagian dari atribut tersebut (Thalheim, 2013). Analoginya, suatu atribut Y dikatakan memiliki dependensi sepenuhnya terhadap X apabila Y memiliki dependensi fungsional terhadap X dan Y tidak memiliki dependensi terhadap bagian dari X, seperti contoh yang sudah dibahas sebelumnya. Contoh lainnya, mari perhatikan tabel 3.2.2 berikut.

Contoh:

Tabel 3.3: Contoh Tabel yang memiliki Dependensi Sepenuhnya

No Pegawai	Nama Pegawai	Jenis Kelamin	Pendidikan	Tahun lulus
P-001	Ana	Perempuan	S1	1987
P-001	Ana	Perempuan	S2	1990
P-002	Risky	Laki-laki	S1	1988
P-002	Risky	Laki-laki	S2	1990
P-002	Risky	Laki-laki	S3	1999
P-003	Lala	Perempuan	S1	1994

Dependensi sepenuhnya terjadi apabila atribut yang memberikan atau pemberi pengaruh lebih dari satu, artinya kita harus melakukan pengecekan secara seksama manakah atribut yang memberikan pengaruh lebih dari satu. Pada tabel 3.3, dapat dilihat jika Tahun lulus bergantung pada No Pegawai dan Pendidikan, atau No Pegawai dan Pendidikan memengaruhi Tahun lulus. Tahun lulus tidak hanya ditentukan oleh Pendidikan saja, melainkan juga No Pegawai, karena dapat dilihat jika satu nilai di No Pegawai memiliki dua bahkan tiga nilai di Tahun lulus, begitupun pada Pendidikan. Pendidikan S1, tidak hanya memiliki Tahun lulus 1987, melainkan juga ada 1988 dan 1994. Artinya disini adalah, Tahun lulus memiliki ketergantungan sepenuhnya terhadap No Pegawai dan Pendidikan, atau sederhananya Tahun lulus tidak dapat ditentukan hanya melalui No Pegawai saja melainkan harus dilihat juga Pendidikannya. Notasinya $\{No\ Pegawai, Pendidikan\} \rightarrow Tahun\ lulus$.

3.2.3 Dependensi Parsial

Suatu atribut Y dapat dikatakan memiliki dependensi atau ketergantungan parsial atau sebagian terhadap X, apabila Y adalah atribut non-primary key dan X adalah primary key (Thalheim, 2013). Y memiliki ketergantungan terhadap sebagian dari X, bukan terhadap keseluruhan dari X. Contohnya, mari perhatikan tabel yang sama seperti contoh kasus ketergantungan sepenuhnya.

Contoh:

Tabel 3.4: Contoh Tabel yang memiliki Dependensi Parsial

No Pegawai	Nama Pegawai	Jenis Kelamin	Pendidikan	Tahun lulus
P-001	Ana	Perempuan	S1	1987
P-001	Ana	Perempuan	S2	1990
P-002	Risky	Laki-laki	S1	1988
P-002	Risky	Laki-laki	S2	1990
P-002	Risky	Laki-laki	S3	1999
P-003	Lala	Perempuan	S1	1994

Pada tabel tersebut dapat dilihat relasi No Pegawai dan Pendidikan merupakan primary key. Mengapa? Suatu relasi dapat dikatakan sebagai primary key, jika atribut dalam relasi tersebut menentukan nilai dari atribut lain. Atribut No Pegawai dan Pendidikan merupakan primary key karna menentukan nilai dari Tahun lulus, atau sederhananya Tahun lulus memiliki ketergantungan terhadap No Pegawai dan Pendidikan. Lalu, di manakah letak dependensi parsialnya? Letaknya ada pada relasi Jenis Kelamin. Mengapa Jenis Kelamin? Karena, Jenis Kelamin tidak dipengaruhi oleh No Pegawai dan Pendidikan, namun hanya dipengaruhi oleh No Pegawai saja. Begitu juga dengan Nama Pegawai yang hanya dipengaruhi oleh No Pegawai saja, maka dari itu relasi Nama Pegawai dan Jenis Kelamin merupakan relasi yang memiliki ketergantungan parsial.

3.2.4 Dependensi Total

Dependensi atau ketergantungan dapat terjadi dengan analogi berikut, sebuah atribut Y dikatakan memiliki dependensi total terhadap X apabila Y memiliki dependensi fungsional terhadap X, dan X juga memiliki dependensi fungsional terhadap Y (Thalheim, 2013). Jika digambarkan ke dalam notasi, menjadi $X \rightarrow\!\!\!+ Y$. Contoh lebih jelasnya, mari perhatikan tabel x berikut.

Contoh:

Tabel 3.5: Contoh Tabel yang memiliki Dependensi Total

Kode Matakuliah	Nama Matakuliah	Kelas
MK-001	Algoritma	Ruang 2
MK-002	Pemrograman	Ruang 3

MK-003	Basis Data	Ruang 2
MK-004	Jaringan Komputer	Ruang 3

Jika melihat dari tabel 3.5, dependensi total ada pada Kode Matakuliah dan Nama Matakuliah. Setiap satu nilai dari Kode Matakuliah, memiliki ketergantungan terhadap satu nilai dari Nama Matakuliah, begitupula sebaliknya. Atribut dari Kelas tidak dapat dikatakan sebagai dependensi total, karena jika dilihat dari relasinya, satu nilai dari atribut Kelas, tidak semata-mata hanya memiliki ketergantungan terhadap satu nilai pada atribut Kode Matakuliah dan Nama Matakuliah. Kode Matakuliah MK-001 memiliki nilai Ruang 2, namun tidak hanya Matakuliah MK-001 saja yang memiliki nilai terhadap Ruang 2, MK-003 pun memiliki nilai yang sama, yaitu Ruang 2. Hal tersebut menyebabkan Kelas tidak memiliki dependensi total terhadap Kode Matakuliah dan Nama Matakuliah, melainkan relasi antara Kode Matakuliah dengan Nama Matakuliah dapat dikatakan sebagai dependensi total.

3.2.5 Dependensi Transitif

Dependensi atau ketergantungan transitif terjadi umumnya pada tabel hasil relasi, atau kondisi di mana terdapat tiga atribut, misalnya terdapat atribut X Y Z. Atribut Z dikatakan memiliki dependensi transitif terhadap X, apabila Z memiliki dependensi fungsional terhadap Y, dan Y memiliki dependensi fungsional terhadap X (Coronel and Morris, 2019). Jadi, notasinya adalah $X \rightarrow Y \rightarrow Z$. Contoh lebih lanjutnya, mari kita lihat ke tabel 3.2.5 berikut.

Contoh:

Tabel 3.6: Contoh Tabel yang memiliki Dependensi Transitif

Kodebrg	Namabrg	Hargabrg	KodeSup	NamaSup	Kota
BR001	Laptop	Rp. 9.625.000	SP002	PT.XYZ	Dps
BR002	Harddisk	Rp. 700.000	SP002	PT.XYZ	Dps
BR003	Mouse	Rp 50.000	SP001	PT.ABC	Bdg
BR004	Speaker	Rp 100.000	SP003	PT.EFG	Jkt
BR005	Keyboard	Rp. 250.000	SP004	PT.JKL	Mlg

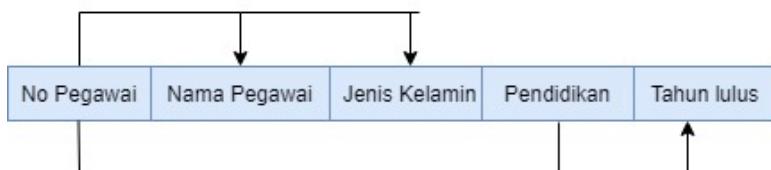
Pada tabel 3.6, dependensi transitif terdapat pada relasi $\text{Kodebrg} \rightarrow \text{NamaSup}$. Mengapa? Karena, pada awalnya dependensi ada pada relasi $\text{Kodebrg} \rightarrow \text{KodeSup}$, lalu $\text{KodeSup} \rightarrow \text{NamaSup}$. Dependensi transitif terjadi karena KodeSup memiliki ketergantungan pada Kodebrg , dan NamaSup

memiliki ketergantungan pada KodeSup. Jadi, jika digambarkan kedalam notasi, menjadi Kodebrg → KodeSup → NamaSup.

3.2.6 Diagram Dependensi Fungsional

Penggambaran hubungan antara penentu dan dependen tidak hanya dapat dinyatakan melalui notasi \rightarrow , namun juga dapat dinyatakan dengan diagram dependensi fungsional (Setiadi, 2017). Berikut merupakan contoh penggambaran relasi dari diagram dependensi fungsional.

Contoh:



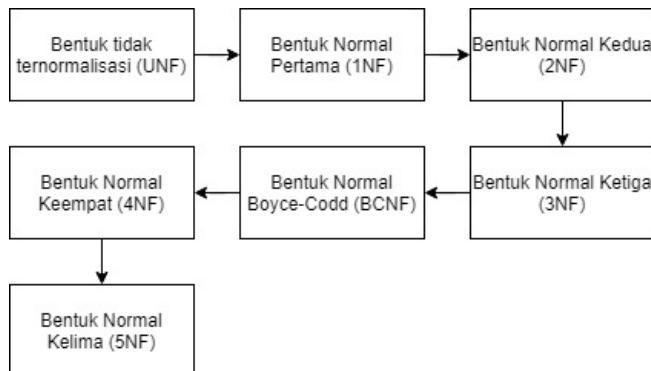
Gambar 3.1: Contoh Penggambaran Relasi Dependensi Fungsional

Gambar 3.1 merupakan contoh relasi atribut PendidikanPegawai. Gambar di atas menjelaskan bahwa No Pegawai menentukan Nama Pegawai dan Jenis Kelamin. No Pegawai dan Pendidikan menentukan Tahun lulus. Jadi, notasinya adalah No Pegawai \rightarrow {Nama Pegawai, Jenis Kelamin}, dan {No Pegawai, Pendidikan} \rightarrow Tahun lulus.

3.3 Normalisasi

Normalisasi merupakan salah satu cara pendekatan atau teknik yang digunakan dalam membangun desain logic database relation dengan menerapkan sejumlah aturan dan kriteria standar, sehingga menghasilkan struktur tabel yang normal atau berstruktur baik (Connoly, Begg and Strachan, 1996). Relasi yang berstruktur baik memiliki kondisi seperti: minim adanya redundansi dan memungkinkan baris-baris dalam relasi disisipkan, dimodifikasi, dan dihapus tanpa menyebabkan kesalahan atau inkonsistensi. Normalisasi memiliki tahapan, tahapannya dimulai dari normalisasi bentuk pertama atau 1NF sampai dengan normalisasi bentuk kelima atau 5NF (Hargo, 2006).

Berikut merupakan langkah-langkah dalam normalisasi.



Gambar 3.2: Langkah-langkah dalam Normalisasi

Gambar 3.2 menjelaskan, tahapan pertama yang dilakukan pada normalisasi adalah menghilangkan atribut yang bernilai ganda dan pengulangan atribut pada grup untuk mendapatkan Bentuk Normal Pertama (1NF). Tahapan selanjutnya adalah menghilangkan ketergantungan parsial dan memastikan fully functional dependency untuk mendapatkan Bentuk Normal Kedua (2NF). Untuk mendapatkan Bentuk Normal Ketiga (3NF), tahapan yang harus dilakukan adalah menghilangkan dependensi transitif. Setelah menghilangkan dependensi transitif, menghilangkan ketergantungan yang penentunya bukan primary key dilakukan untuk mendapatkan Bentuk Normal Boyce-Codd (BCNF). Setelah mendapatkan BCNF, dilanjutkan dengan proses menghilangkan lebih dari satu dependensi berganda. Hal ini bertujuan untuk mendapatkan Bentuk Normal Keempat (4NF). Tahapan akhir untuk mendapatkan Bentuk Normal Kelima adalah dengan mengatasi dependensi gabungan. Sub bab berikut akan membahas tahapan normalisasi dari tahap awal hingga akhir untuk mendapatkan bentuk normal kelima.

3.3.1 Bentuk Normal Pertama (1NF)

Suatu relasi dapat dikatakan telah memenuhi Bentuk Normal Pertama (1NF), apabila setiap atribut dari relasi tersebut hanya memiliki nilai tunggal dan tidak ada pengulangan grup atribut dalam baris atau atomic (Silberschatz, Korth and Sudarshan, 2019). Untuk lebih jelasnya, mari kita perhatikan contoh berikut.

Contoh:

Tabel 3.7: Contoh Tabel yang Belum Memenuhi 1NF

Kodepegawai	Namapegawai	Telepon
BR001	Ana	081234990897 0361242108
BR002	Ani	089709665432 0361225678
BR003	Devi	081338564321
BR004	Risky	085737442821 0361556743
BR005	Indah	087665789099

Pada tabel 3.7 terdapat data dari tabel Pegawai, yang didalamnya terdapat atribut Kodepegawai, Namapegawai dan Telepon. Tabel tersebut belum dapat dikatakan memenuhi Bentuk Normal Pertama (1NF), mengapa? Karena masih adanya atribut yang memiliki nilai lebih dari satu, atau non atomic. Atribut manakah yang disebut sebagai non atomic? Atribut yang nilainya lebih dari satu, yaitu atribut Telepon. Seperti yang bisa dilihat, kode pegawai BR001, BR002, dan BR004, memiliki lebih dari satu nomor telepon. Lantas, bagaimanakah solusinya? Yaitu dengan menambahkan baris untuk memecah nilai yang ada pada atribut Telepon, agar memenuhi Bentuk Normal Pertama (1NF). Tabel 3.8 merupakan tabel yang telah dipecah nilai pada atribut Telepon agar memenuhi Bentuk Normal Pertama (1NF).

Tabel 3.8: Contoh Tabel yang Telah Memenuhi 1NF

Kodepegawai	Namapegawai	Telepon
BR001	Ana	081234990897
BR001	Ana	0361242108
BR002	Ani	089709665432
BR002	Ani	0361225678
BR003	Devi	081338564321
BR004	Risky	085737442821
BR004	Risky	0361556743
BR005	Indah	087665789099

Jika redundansi terjadi dengan kasus mengulang grup yang telah ada, bagaimanakah penyelesaiannya? Mari kita perhatikan tabel 3.9. Tabel 3.9 memiliki redundansi atau pengulangan pada grup atribut Id Penulis dan Nama

Penulis. Kondisi tabel x tidak memenuhi Bentuk Normal Pertama (1NF), untuk dapat menjadi Bentuk Normal Pertama (1NF), maka harus ada penambahan baris untuk menempatkan nilai atribut grup yang redundan. Tabel 3.9 menunjukkan kondisi tabel yang telah memenuhi Bentuk Normal Pertama (1NF).

Contoh:

Tabel 3.9: Contoh Tabel yang Masih Memiliki Redudansi

Kode Jurnal	Tahun Terbit	Id Penulis		Nama Penulis	Id Penulis	Nama Penulis
JUR001	2001	ID999		Ana	ID998	Ani
JUR002	2002	ID998		Ani		
JUR003	2002	ID997		Kelly	ID999	Ana
JUR004	2003	ID996		Devi	ID997	Kelly
JUR005	2002	ID994		Risky		

Tabel 3.10: Contoh Tabel yang Telah Memenuhi 1NF

Kode Jurnal	Tahun Terbit	Id Penulis		Nama Penulis
JUR001	2001	ID999		Ana
JUR001	2001	ID998		Ani
JUR002	2002	ID998		Ani
JUR003	2002	ID997		Kelly
JUR003	2002	ID999		Ana
JUR004	2003	ID996		Devi
JUR004	2003	ID997		Kelly
JUR005	2002	ID994		Risky

3.3.2 Bentuk Normal Kedua (2NF)

Suatu relasi dikatakan memenuhi Bentuk Normal Kedua (2NF), apabila relasi tersebut telah memenuhi Bentuk Normal Pertama (1NF) dan setiap atribut yang bukan kunci utama tergantung secara fungsional terhadap semua atribut primary key (Connoly, Begg and Strachan, 1996). Bukan hanya sebagian atribut primary key atau harus dependensi fungsional sepenuhnya. Contohnya adalah sebagai berikut

Contoh:

1. Diketahui tabel X=(A,B,C,D,E)
2. Dengan kondisi A,B merupakan primary key
3. Dengan notasi dependensi fungsional $A,B \rightarrow C,D,E$. Dapat disimpulkan C,D,E memiliki dependensi fungsional terhadap primary key A,B, maka tabel X telah memenuhi kondisi Bentuk Normal Kedua (2NF). Berikut penjelasan lebih detailnya:

$A,B \rightarrow C,D,E$ berarti:

$A,B \rightarrow C$,

$A,B \rightarrow D$, dan

$A,B \rightarrow E$

4. Jadi, semua atribut yang bukan merupakan primary key memiliki dependensi penuh terhadap A,B. Contoh selanjutnya, mari kita perhatikan tabel berikut.

Contoh:

Tabel 3.11: Contoh Tabel yang Belum Memenuhi 2NF

NIM	Matakuliah	Biaya	Kelas
1204505019	Basisdata	200000	A
1204505020	Jaringan	250000	A
1204505019	Jaringan	250000	B
1204505018	Algoritma	100000	A
1204505016	Basisdata	200000	B

Tabel 3.11 merupakan tabel Pembayaran Matakuliah, yang mana diketahui Pembayaran Matkul=(NIM,Matakuliah,Biaya,Kelas). Primary key ada pada atribut NIM dan Matakuliah, sedangkan atribut Biaya memiliki dependensi fungsional pada atribut Matakuliah. Kondisi tabel ini masih belum memenuhi Bentuk Normal Kedua (2NF). Mengapa? Karena atribut yang bukan primary key, yaitu Biaya dan Grade tidak memiliki dependensi fungsional terhadap atribut NIM dan Matakuliah. Atribut Biaya hanya memiliki dependensi fungsional terhadap Matakuliah. Maka dari itu, yang perlu dilakukan agar dapat memenuhi Bentuk Normal Kedua (2NF), adalah dengan membuat tabel

baru PembayaranMatkul1=(NIM,Matakuliah,Kelas) dan PembayaranMatkul2=(Matakuliah,Biaya). Sehingga, hasilnya seperti ini.

Tabel 3.12: Tabel Pembayaran Matkul 1 (Memenuhi 2NF)

NIM	Matakuliah	Kelas
1204505019	Basisdata	A
1204505020	Jaringan	A
1204505019	Jaringan	B
1204505018	Algoritma	A
1204505016	Basisdata	B

Tabel 3.13: Tabel Pembayaran Matkul 2 (Memenuhi 2NF)

Matakuliah	Biaya
Basisdata	200000
Jaringan	250000
Algoritma	100000

3.3.3 Bentuk Normal Ketiga (3NF)

Suatu relasi dikatakan telah memenuhi Bentuk Normal Ketiga (3NF), apabila relasi tersebut telah memenuhi Bentuk Normal Kedua (2NF) dan atribut yang bukan primary key tidak memiliki dependensi transitif terhadap primary key (Coronel and Morris, 2019). Berikut adalah contoh dari tabel yang belum memenuhi Bentuk Normal Ketiga (3NF).

Contoh:

1. Diketahui tabel X=(A,B,C,D,E)
2. Dengan kondisi A,B merupakan primary key,
3. Memiliki dependensi fungsional $A,B \rightarrow C,D,E$ dan $C \rightarrow D,E$. Berikut untuk penjabaran notasinya

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

A,B merupakan primary key.

Karena sifat refleksif maka $A,B \rightarrow A,B$.

Sehingga $A,B \rightarrow A,B,C,D,E$. Kondisi ini memenuhi Bentuk Normal Kedua (2NF)

Diketahui $C \rightarrow D, E$. Karena sifat refleksif, maka $C \rightarrow C$. Sehingga $C \rightarrow C, D, E$, dan C bukan merupakan primary key.

- Kondisi tabel X tidak memenuhi Bentuk Normal Ketiga, karena D,E memiliki dependensi fungsional terhadap C, yang mana C bukan merupakan primary key.

Jadi, bagaimanakah solusinya? Agar tabel X memenuhi Bentuk Normal Ketiga(3NF), maka perlu dilakukan dekomposisi menjadi:

$X_1 = (A, B, C)$ dan $X_2 = (C, D, E)$, sehingga tabel X_1 dan X_2 memenuhi Bentuk Normal Ketiga (3NF). Contoh lainnya, mari perhatikan tabel order barang berikut.

Contoh:

Tabel 3.14: Tabel Order Barang (Memenuhi 2NF)

No order	Tgl order	Item	Deskripsi
1001	20-03-10	P1	Mouse
1001	20-03-10	P2	Keyboard
1001	20-03-10	P3	Speaker
1001	20-03-10	P4	Flashdisk
1002	21-03-10	P1	Mouse
1002	21-03-10	P2	Keyboard
1003	22-03-10	P1	Mouse
1003	22-03-10	P4	Flashdisk

Tabel order barang di atas sudah memenuhi 2NF, tetapi belum memenuhi kondisi Bentuk Normal Ketiga (3NF). Mengapa? Karena berdasarkan tabel tersebut, ada dependensi antara atribut pada Deskripsi dengan Item. Hal ini tidak memenuhi 3NF karena Deskripsi hanya memiliki dependensi terhadap Item, sedangkan tidak dengan No order dan Tgl order, padahal No order dan Tgl order merupakan primary key dari tabel order tersebut. Maka dari itu, untuk memenuhi Bentuk Normal Ketiga (3NF), perlu dilakukan dekomposisi tabel order menjadi notasi Tabel order(No order,Tgl order,Item) dan Tabel Item(Item,Deskripsi). Berikut penjabarannya:

Tabel 3.15: Tabel Oder (Memenuhi 3NF)

No order	Tgl order	Item
1001	20-03-10	P1
1001	20-03-10	P2
1001	20-03-10	P3
1001	20-03-10	P4
1002	21-03-10	P1
1002	21-03-10	P2
1003	22-03-10	P1
1003	22-03-10	P4

Tabel 3.16: Tabel Item (Memenuhi 3NF)

Item	Deskripsi
P1	Mouse
P2	Keyboard
P3	Speaker
P4	Flashdisk

3.3.4 Bentuk Normal Boyce-Codd (BCNF)

Secara praktis, tujuan rancangan database adalah cukup sampai pada level 3NF. Namun, untuk beberapa kondisi tertentu, bisa menerapkan BCNF jika ingin mendapatkan rancangan yang lebih baik lagi. BCNF ditemukan oleh R.F. Boyce dan E.F. Codd (Silberschatz, Korth and Sudarshan, 2011). Suatu relasi dapat dikatakan memenuhi Bentuk Normal Boyce-Codd (BCNF), apabila setiap atribut primary key pada suatu relasi adalah candidate key dan tidak ada atribut non primary key yang memiliki dependensi terhadap atribut non primary key juga. Candidate key adalah atribut-atribut dari entitas yang mungkin dapat digunakan sebagai primary key. BCNF hampir sama dengan 3NF, dengan kata lain setiap BCNF adalah 3NF. Berikut adalah contohnya.

Contoh:

Tabel 3.17: Tabel Pemilihan MK (Memenuhi 1NF)

Student_id	Subject	Professor
101	Algoritma	P.Algoritma
101	Basisdata	P.Basisdata
102	Algoritma	P.Algoritma2
103	Basisdata lanjut	P.BasisdataLanjut
104	Algoritma	P.Algoritma

Tabel di atas merupakan tabel Pemilihan MK, yang mana student_id dan subject merupakan primary key, dan atribut Professor memiliki dependensi terhadap atribut Subject. Notasinya adalah

$\text{Student_id}, \text{Subject} \rightarrow \text{Professor}$ (Student_id dan Subject merupakan primary key)

$\text{Professor} \rightarrow \text{Subject}$ (Professor merupakan atribut non primary key dan Subject merupakan atribut primary key)

Tabel ini telah memenuhi Bentuk Kondisi Pertama (1NF), karena seluruh atribut yang ada pada tabel non primary key, yaitu tabel Professor telah memiliki dependensi fungsional terhadap atribut primary key. Namun, yang membuat tabel ini belum memenuhi kondisi BCNF adalah notasi $\text{Professor} \rightarrow \text{Subject}$, yang mana terdapatnya atribut primary key yang memiliki dependensi terhadap atribut non primary key. Bagaimana solusi yang dapat dilakukan? Solusi yang dapat dilakukan adalah dengan memecah tabel tersebut menjadi dua bagian, yaitu Student_table(student_id, prof_id) dan Professor_table(prof_id, professor dan subject). Mengapa ada atribut baru, yaitu Prof_id? Karena, atribut Subject dikatakan memiliki dependensi terhadap atribut Professor, namun atribut Professor bukanlah primary key. Maka dari itu, perlu untuk dibuatkan primary key pada tabel Professor agar atribut Subject tetap dapat memiliki dependensi terhadap tabel Professor. Berikut adalah penyelesaiannya.

Tabel 3.18: Student Table (Memenuhi BCNF)

Student_id	Prof_id
101	p01
101	p01
102	p02

103	p03
104	p04

Tabel 3.19: Professor Table (Memenuhi BCNF)

Prof_id	Professor
p01	P.Algoritma
p01	P.Basisdata
p02	P.Algoritma2
p03	P.BasisdataLanjut
p04	P.Algoritma

3.3.5 Bentuk Normal Keempat (4NF)

Bentuk Normal Keempat (4NF) terjadi apabila adanya dependensi multi nilai terjadi dalam relasi apapun. Apakah yang dimaksud dengan dependensi multi nilai? Sebuah tabel dikatakan memiliki dependensi multi nilai, jika:

1. Misalkan terdapat notasi $A \rightarrow B$, jika untuk satu nilai A terdapat beberapa nilai B, maka tabel mungkin memiliki dependensi multi nilai.
2. Tabel harus memiliki setidaknya 3 kolom untuk memiliki ketergantungan multi nilai.
3. Untuk relasi $X(A,B,C)$, jika ada dependensi multi nilai antara A, dan B, maka B dan C harus independen satu sama lain.

Untuk lebih jelasnya, mari kita perhatikan contoh dari tabel matakuliah berikut.

Contoh:

Tabel 3.20: Tabel Matakuliah (Belum Memenuhi 4NF)

Student_id	Subject	Alamat
101	Algoritma	Denpasar
101	Basisdata	Badung
102	Pemrograman	Denpasar
102	Basisdata lanjut	Badung

Tabel di atas merupakan tabel matakuliah yang memiliki atribut student_id, subject dan alamat. Student_id 101 memilih 2 dua subject pada atribut Subject,

yaitu algoritma dan basisdata, dan memiliki dua alamat, yaitu Denpasar dan Badung. Dalam kondisi ini, student_id 101 memiliki dua catatan, yang mana kondisi ini akan menimbulkan dua catatan lagi. Hal ini disebabkan karena untuk satu student_id yang memiliki dua alamat bersamaan dengan dua subject, maka alamat yang dimiliki tersebut harus ditentukan secara spesifik seperti tabel 3.21 di bawah ini.

Tabel 3.21: Tabel Matakuliah (Belum Memenuhi 4NF)

Student id	Subject	Alamat
101	Algoritma	Denpasar
101	Basisdata	Badung
101	Algoritma	Badung
101	Basisdata	Denpasar

Pada tabel 3.21, tidak ada hubungan antara kolom subject dan alamat, maka subject dan alamat dapat dikatakan independen satu sama lain. Lalu, bagaimanakah cara untuk memenuhi kondisi Bentuk Normal Keempat (4NF)? Tabel x dapat diuraikan menjadi 2 tabel, yaitu tabel Subject dan Alamat. Berikut adalah contohnya.

Tabel 3.22: Tabel Subject (Memenuhi 4NF)

Student id	Subject
101	Algoritma
101	Basisdata
102	Pemrograman
102	Basisdata lanjut

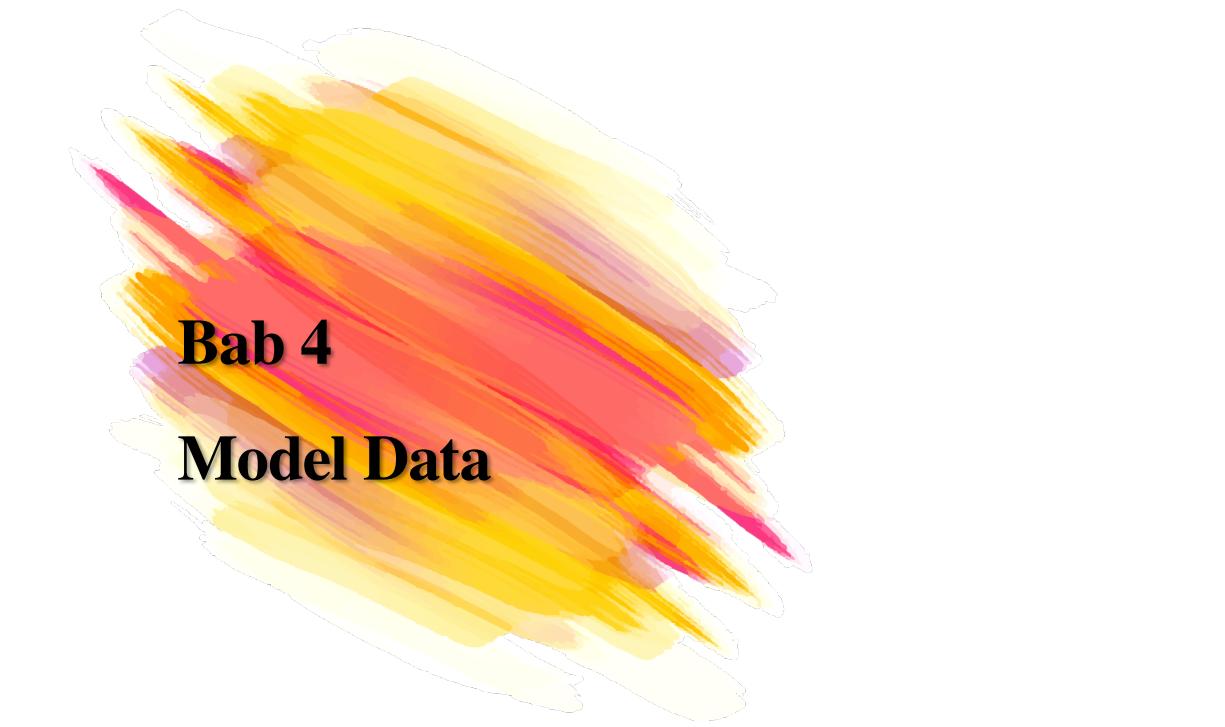
Tabel 3.22: Tabel Alamat (Memenuhi 4NF)

Student id	Alamat
101	Denpasar
101	Badung
102	Denpasar
102	Badung

Saat ini, hubungan relasi di tabel tersebut memenuhi Bentuk Normal Keempat (4NF). Sebuah tabel dapat memiliki dependensi fungsional bersama dengan dependensi multi nilai. Kolom yang memiliki dependensi secara fungsional dan kolom dependen multi nilai dipindahkan ke tabel terpisah.

3.3.6 Bentuk Normal Kelima (5NF)

Suatu tabel dikatakan memenuhi kondisi Bentuk Normal Kelima (5NF), jika sudah tidak dapat didekomposisi menjadi sejumlah tabel lebih kecil lagi. Bentuk Normal Kelima (5NF) berdasarkan pada konsep dependensi gabungan (join dependencies), yang berarti bahwa sebuah tabel setelah didekomposisi menjadi 3 atau lebih tabel kecil, harus dapat digabungkan kembali untuk terbentuk menjadi tabel awal (Silberschatz, Korth and Sudarshan, 2019). Kesimpulan yang dapat ditarik adalah suatu tabel dapat disebut sebagai Bentuk Normal Kelima (5NF) ketika tabel tersebut tidak dapat didekomposisi atau dipecah lagi.



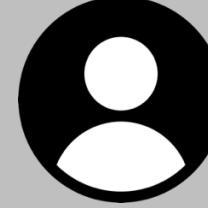
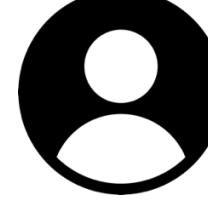
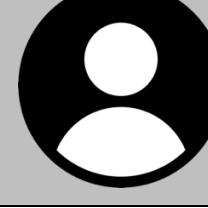
Bab 4

Model Data

4.1 Pengertian Model Data

Diketahui kampus STIKI memiliki banyak mahasiswa. Di antaranya adalah Jono, Santi, Agus, dan Budi. Jono adalah mahasiswa dengan nim 17101220. Jono mengambil jurusan Komputer Akuntansi dan Bisnis dan tinggal di Denpasar. Makanan kesukaan Jono adalah bakso. Santi adalah mahasiswa dengan nim 17101952. Santi mengambil jurusan Desain Grafis dan Multimedia dan tinggal di Denpasar. Santi suka travelling dan membaca novel. Agus adalah mahasiswa dengan nim 17101880. Agus mengambil jurusan Komputer Akuntansi dan Bisnis dan tinggal di Gianyar. Hobby Agus adalah bermain futsal. Budi adalah mahasiswa dengan nim 17101908. Agus mengambil jurusan Desain Grafis dan Multimedia dan tinggal di Denpasar. Hobby Budi adalah travelling dan bermain futsal. Jika data mahasiswa yang ada di kampus STIKI akan disimpan ke dalam basis data, bagaimana data-data ini disusun di dalam basis data?

Sebelum membayangkan bagaimana data-data ini disusun ke dalam basis data (di dalam komputer), mari mendeskripsikan data-data tersebut ke dalam kertas agar mudah dipahami. Ada beberapa cara yang dapat digunakan, misalnya dengan dengan mengelompokkan data per mahasiswa seperti yang dideskripsikan dalam Gambar 4.1 atau menggunakan tabel seperti yang dideskripsikan dalam Tabel 4.1.

	NIM : 17101220 Nama : Jono Jurusan : Komputer Akuntansi dan Bisnis Alamat : Denpasar Hobby : -
	NIM : 17101952 Nama : Santi Jurusan : Desain Grafis dan Multimedia Alamat : Denpasar Hobby : <i>travelling</i> , membaca novel
	NIM : 17101880 Nama : Agus Jurusan : Komputer Akuntansi dan Bisnis Alamat : Denpasar Hobby : bermain futsal
	NIM : 17101908 Nama : Budi Jurusan : Desain Grafis dan Multimedia Alamat : Denpasar Hobby : <i>travelling</i> , bermain futsal

Gambar 4.1: Deskripsi dengan mengelompokkan data per mahasiswa

Tabel 4.1: Deskripsi menggunakan tabel

NIM	Nama	Jurusan	Alamat	Hobby
17101220	Jono	Komputer Akuntansi dan Bisnis	Denpasar	-
17101952	Santi	Desain Grafis dan Multimedia	Denpasar	travelling, membaca novel
17101880	Agus	Komputer Akuntansi dan Bisnis	Gianyar	bermain futsal
17101908	Budi	Desain Grafis dan Multimedia	Denpasar	travelling, bermain futsal

Gambar 4.1 dan Tabel 4.1 merupakan contoh cara yang dapat digunakan untuk mendeskripsikan data. Untuk mendeskripsikan data yang akan disimpan ke dalam basis data digunakan model data. Model data didefinisikan sebagai representasi yang relatif sederhana, biasanya dalam bentuk grafis, dari struktur data dunia nyata yang jauh lebih kompleks (Coronel and Morris, 2016).

Jika membandingkan deskripsi data yang ada di dunia nyata dengan deskripsi dalam Gambar 4.1 atau Tabel 4.1, memahami data yang dideskripsikan dalam Gambar 4.1 atau Tabel 4.1 tentunya lebih mudah. Deskripsi dalam Gambar 4.1 atau Tabel 4.1 dapat membantu memahami data secara lebih mudah sehingga dapat disebut sebagai alat bantu. Demikian juga halnya dengan model data, model data dapat membantu memahami data. Model data dapat didefinisikan sebagai kumpulan alat bantu secara konseptual untuk dapat memahami data, relasi antar data, semantik data, dan *consistency constraint* (Silberschatz, Korth and Sudarshan, 2019)

4.2 Arti Penting Model Data

Pembuatan basis data dapat dianalogikan dengan pembangunan rumah. Pembangunan rumah tentunya tidak dapat dilakukan seorang diri. Ada pemilik rumah, ada arsitek yang akan membuat rancangan rumah sesuai dengan keinginan pemilik, ada tukang yang akan melaksanakan pembangunan rumah. Agar pembangunan rumah dapat berjalan dengan lancar, sebuah rancangan

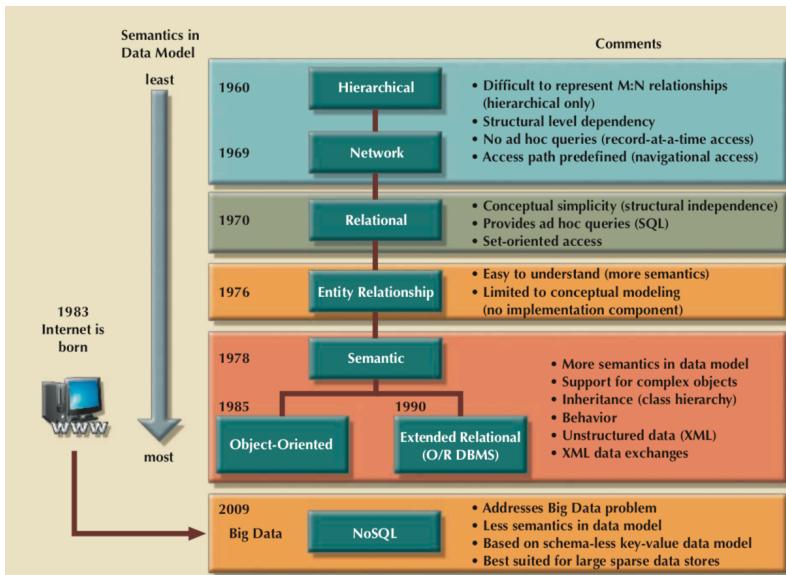
atau blueprint rumah wajib dibuat terlebih dahulu. Tanpa adanya blueprint rumah, rawan sekali muncul konflik di dalam proses pembangunan rumah.

Seperti pembangunan rumah yang melibatkan beberapa pihak, pengembangan aplikasi juga melibatkan beberapa pihak seperti designer, programmer, dan juga end user. Masing-masing user memiliki kepentingan dan sudut pandang yang berbeda mengenai data-data di dalam basis data. Seorang programmer lebih berkepentingan dengan lokasi penyimpanan data, format data, dan requirement untuk laporan-laporan tertentu. End user yang berada pada top level tidak berkepentingan dengan data secara mendetail. Mereka lebih berkepentingan dengan data secara universal. End user pada divisi tertentu berkepentingan dengan data-data yang terkait dengan divisinya saja. Model data membantu memfasilitasi interaksi di antara pihak-pihak yang berkepentingan dengan basis data. Model data yang dikembangkan dengan baik bahkan dapat meningkatkan pemahaman kita akan organisasi di mana basis datanya hendak dibangun.

Model data, seperti halnya blueprint rumah, merupakan sebuah abstraksi. Sebuah blueprint rumah tentunya tidak dapat ditinggali. Agar dapat ditinggali, blueprint tersebut harus diwujudkan menjadi rumah terlebih dahulu. Demikian juga halnya dengan model data. Model data tidak dapat digunakan untuk menyimpan atau mengambil data. Namun bukan berarti model data tidak berguna. Model data, sebagaimana *blueprint* rumah yang membantu dalam membangun sebuah rumah, memiliki peran yang penting untuk membangun basis data yang baik.

4.3 Evolusi Model Data

Model data yang dikenal saat ini dihasilkan melalui proses yang panjang, seperti yang dideskripsikan dalam Gambar 4.2. Mulai dari tahun 1960-an di mana model data Hierarki dan model data Network muncul. Pada tahun 1970, muncullah model data Relasional, yang hingga kini masih digunakan secara luas. Pada tahun 1976, muncullah model data Entity Relationship, yang kini digunakan sebagai komplemen dari model data Relasional. Pada pertengahan tahun 1980-an, muncullah ide model data berbasis semantik yaitu model data berorientasi objek dan model data Object/ Relational. Dan pada tahun 2000-an, muncullah model data yang dapat mendukung Big Data.

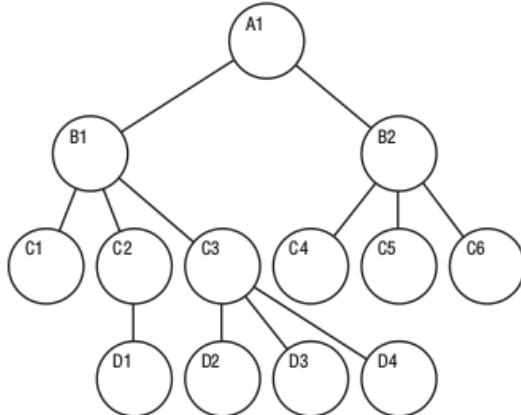


Gambar 4.2: Perkembangan model data (Coronel and Morris, 2016)

4.3.1 Model Hierarki dan Network

Model data Hierarki dikembangkan pada tahun 1960-an. Model data Hierarki merupakan jawaban atas pertanyaan bagaimana membuat program komputer yang dapat digunakan untuk melacak jutaan suku cadang dari sebuah roket. Saat itu, Amerika Serikat tengah membuat Project Apollo yang memiliki tujuan untuk mendaratkan manusia ke Bulan. Pada model data Hierarki, data diatur sedemikian rupa sehingga menyerupai bentuk pohon (tree). Data disimpan dalam record dan dihubungkan dengan record lainnya dengan menggunakan link. Record sendiri merupakan kumpulan dari field. Sebuah field hanya dapat menyimpan satu buah nilai.

Tree tersusun dari tingkatan-tingkatan yang disebut dengan level atau segment. Level yang berada di atas disebut dengan parent, sedangkan level yang berada di bawahnya disebut dengan *child*. Parent dapat memiliki banyak child, akan tetapi child hanya dapat memiliki satu buah parent. Parent yang berada paling atas, yang tidak memiliki parent lagi, disebut dengan root. Pemodelan data dalam bentuk tree dapat dilihat pada Gambar 4.3.



Gambar 4.3: Pemodelan data dalam bentuk tree (MariaDB, 2015a)

Penggambaran model data Hierarkhi dengan menggunakan model data Relasional dideskripsikan dalam Gambar 4.4. Tabel Employee merupakan parent dari tabel Computer. Seorang pekerja, misalnya Brad, diberikan dua komputer yaitu PC dengan serial number A100001 dan laptop dengan serial A200002. Ketika komputer dengan serial number A100001 diberikan kepada Brad, komputer tersebut tidak dapat berikan kepada pekerja selain Brad karena komputer tersebut sudah diberikan kepada Brad.

Tabel Employee:

ID_Pegawai	Nama Pegawai	Divisi	Gaji
1	Brad	Marketing	60000
2	Jerry	Programmer	15000
3	July	Secretary	70000

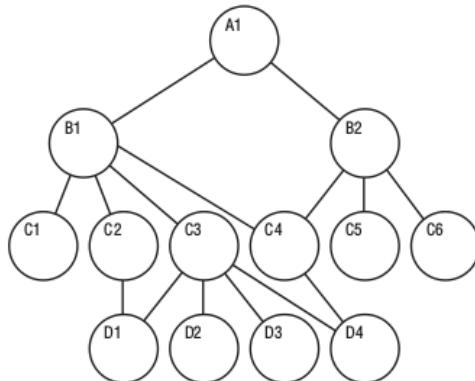
Tabel Computer:

Serial_Number	Tipe	ID_Pegawai
A100001	PC	1

A200002	laptop	1
C100002	PC	2

Gambar 4.4: Ilustrasi Model Data Hierarkhi menggunakan Basis Data Relasional

Model data Network merupakan pengembangan dari model data Hierarki. Model data ini mengatasi kelemahan yang terdapat pada model data Hierarki, yaitu child hanya dapat memiliki satu parent. Pemodelan data dalam bentuk Network dideskripsikan dalam Gambar 4.5.



Gambar 4.5: Pemodelan data dalam bentuk network (MariaDB, 2015b)

Meskipun Model Data Network lebih fleksibel dibandingkan dengan Model Data Hierarkhi, implementasi dan *maintenance model* data ini sulit untuk dilakukan. Programmer harus memahami struktur data untuk dapat membuat model yang efien. Selain itu, ketiadaan *ad hoc query* mengakibatkan beban kerja terpusat pada programmer, misalnya untuk membuat laporan-laporan tertentu.

4.3.2 Model Relasional (Relational)

Model data relasional diperkenalkan oleh E.F. Codd pada tahun 1970. Model ini didasarkan pada konsep matematika yaitu relasi. Saat ini, model data Relasional masih menjadi model data yang paling banyak digunakan dalam aplikasi pemrosesan data komersial.

Tabel disusun oleh baris-baris, di mana satu baris merepresentasikan satu buah informasi. Ada empat mahasiswa yang diceritakan dalam Sub Bab 4.1. Masing-masing mahasiswa akan menempati satu baris di dalam tabel sehingga terdapat empat baris di dalam tabel. Mahasiswa memiliki lima buah properti, yaitu NIM, nama, jurusan, alamat, dan hobby.

Misalkan di STIKI terdapat dosen yang bernama Eddy, dengan NIDN 0810109001 dan Kadek, dengan NIDN 0801039202. Seluruh mahasiswa yang ada di STIKI memiliki dosen pembimbing akademik. Dosen pembimbing Jono dan Agus adalah Eddy, dosen pembimbing Santi dan Budi adalah Kadek. Di sini dapat dilihat entitas-entitas yang ada di himpunan entitas Mahasiswa memiliki hubungan dengan dengan entitas-entitas yang ada di himpunan entitas Dosen.

Informasi mengenai dosen yang dideskripsikan di atas dapat disimpan ke dalam tabel Dosen. Ada dua baris yang terdapat di dalam tabel dosen yaitu baris yang akan digunakan untuk menyimpan informasi Eddy dan baris yang akan digunakan untuk menyimpan informasi Kadek. Tabel Dosen dideskripsikan dalam Tabel 4.2.

Tabel 4.2: Tabel dosen

NIDN	Nama
0810109001	Eddy
0801039202	Kadek

Deskripsi di atas menceritakan pembimbing akademik Eddy adalah pembimbing akademik dari Jono dan Agus. Bagaimana cara mendeskripsikan hal ini menggunakan model data Relasional? Relasi di dalam model data Relasional dideskripsikan dengan menggunakan kolom. Kolom NIDN dapat ditambahkan ke dalam tabel mahasiswa sehingga tabel mahasiswa dapat dihubungkan dengan tabel Dosen. Tabel Mahasiswa dideskripsikan dalam Tabel 4.3.

Tabel 4.3: Tabel mahasiswa

NIM	Nama	Jurusan	Alamat	Hobby	NIDN
17101220	Jono	Komputer Akuntansi dan Bisnis	Denpasar	-	0810109001
17101952	Santi	Desain Grafis dan Multimedia	Denpasar	travelling, membaca novel	0801039202
17101880	Agus	Komputer Akuntansi dan Bisnis	Gianyar	bermain futsal	0810109001
17101908	Budi	Desain Grafis dan Multimedia	Denpasar	travelling, bermain futsal	0801039202

Pertemuan dari baris dan kolom akan berisi nilai kolom dari suatu informasi. Nilai tersebut tidaklah tanpa aturan. Ada aturan, nilai seperti apakah yang dapat menjadi nilai dari suatu kolom. Nilai-nilai yang dapat menjadi nilai kolom disebut dengan domain. Sebagai contoh, perhatikan kolom NIM dari tabel yang ada di Tabel 4.3. NIM hanya disusun dari kombinasi angka sebanyak delapan digit, sehingga tidak akan mahasiswa yang memiliki NIM „satu“ atau „pertamax“, yang disusun dari kombinasi karakter (bukan dari kombinasi angka sebanyak delapan digit).

Nilai dari suatu kolom atau kombinasi nilai dari beberapa kolom dapat digunakan untuk mengidentifikasi suatu baris. Kombinasi dari kolom yang dapat digunakan untuk mengidentifikasi suatu baris disebut dengan superkey. Sebagai contoh, kombinasi kolom NIM dan nama pada tabel mahasiswa dapat digunakan untuk mengidentifikasi suatu baris. Demikian juga dengan kombinasi nama dan alamat (dengan asumsi tidak mungkin ada dua orang dengan nama yang sama tinggal di satu rumah). Kombinasi NIM dan nama merupakan superkey, demikian juga dengan kombinasi nama dan alamat.

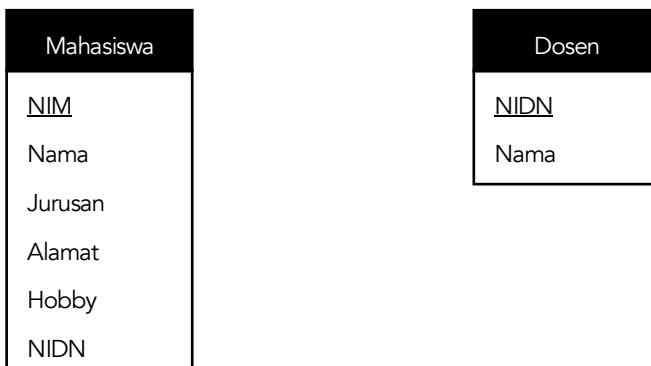
Superkey dapat digunakan untuk mengidentifikasi suatu baris, akan tetapi kombinasi dari kolom-kolom penyusunnya belum tentu minimal. Sebagai contoh, kombinasi dari NIM dan nama merupakan superkey. Jika kolom nama diabaikan, ternyata NIM saja dapat digunakan untuk mengidentifikasi suatu baris (karena tidak ada dua mahasiswa atau lebih yang memiliki NIM yang sama). Nama dan alamat merupakan superkey. Namun jika salah satu dihilangkan, kolom yang tersisa tidak dapat lagi digunakan untuk mengidentifikasi suatu baris (bisa jadi ada beberapa orang yang memiliki nama

yang sama). Superkey yang disusun oleh kombinasi kolom minimal disebut dengan candidate key.

NIM adalah candidate key di dalam tabel Mahasiswa. Kombinasi kolom nama dan alamat juga merupakan candidate key. Kedua candidate key ini dapat digunakan untuk mengidentifikasi baris-baris di dalam tabel Mahasiswa. Salah satu dari candidate key ini akan dipilih menjadi primary key. Sebuah tabel hanya dapat memiliki satu primary key.

Misalkan kolom yang dipilih sebagai primary key pada tabel Mahasiswa adalah NIM dan kolom yang dipilih sebagai primary key pada tabel Dosen adalah NIDN. Pada tabel Mahasiswa terdapat kolom NIDN juga, di mana nilai yang dapat diterima hanyalah nilai-nilai yang sudah didefinisikan di dalam kolom NIDN di tabel Dosen. Kolom atau kombinasi kolom yang mengacu ke primary key tabel lain disebut dengan foreign key. Kolom NIDN di tabel Mahasiswa merupakan foreign key yang mengacu ke primary key dari tabel Dosen.

Struktur dari model data relasional dapat dideskripsikan dengan menggunakan diagram skema. Diagram skema memuat nama tabel dan kolom-kolom dari tabel tersebut. Kolom atau kombinasi kolom yang menjadi primary key dituliskan dengan menambahkan garis di bawah nama kolom. Foreign key digambarkan dengan menggunakan tanda panah menuju primary key yang diacu. Contoh dari diagram skema dideskripsikan pada Gambar 4.6.



Gambar 4.6: Contoh diagram skema

4.3.3 Model Entity-Relationship

Model data Relasional dapat diterima dengan luas karena kesederhanaan konsepnya. Akan tetapi, masalah muncul ketika requirement yang diberikan semakin kompleks. Requirement yang semakin kompleks menghasilkan model yang makin kompleks juga, dan ternyata hal ini sulit dilakukan dengan menggunakan model data Relasional.

Model data Entity-Relationship diperkenalkan pada tahun 1976 oleh Peter Chen. Model ini dapat diterima dengan mudah karena data-datanya direpresentasikan dalam bentuk grafis dan Model data Entity-Relationship ini dapat dipadukan dengan model data Relasional.

Komponen dari model data Entity Relationsip terdiri dari:

1. Entitas

Entitas merupakan sesuatu atau objek yang ada di dunia nyata yang dapat dibedakan dengan objek lainnya. Sebagai contoh, perhatikan deskripsi permasalahan yang ada pada Sub Bab 4.1. Ada empat entitas yang ada berdasarkan permasalahan yang dideskripsikan dalam Sub Bab 4.1, yaitu mahasiswa yang bernama Jono, mahasiswa yang bernama Santi, mahasiswa yang bernama Agus, dan mahasiswa yang bernama Budi. Santi dapat kita bedakan dengan Budi, sehingga Santi dan Budi masing-masing merupakan sebuah entitas.

Keempat entitas yang dideskripsikan dalam Sub Bab 4.1 memiliki karakteristik yang sama. Semuanya punya NIM, nama, jurusan, alamat, dan hobby (kecuali Jono yang hobinya tidak diketahui). Nilainya bisa jadi berbeda, akan tetapi keempat entitas mahasiswa ini memiliki karakteristik yang sama, yaitu sama-sama memiliki NIM, nama, jurusan, alamat, dan hobby. Entitas-entitas yang memiliki karakteristik yang sama dapat dikelompokkan menjadi himpunan entitas (entity set). Entitas Jono, Santi, Agus, dan Budi memiliki karakteristik yang sama sehingga dapat dikumpulkan ke dalam himpunan entitas, sebut saja Mahasiswa.

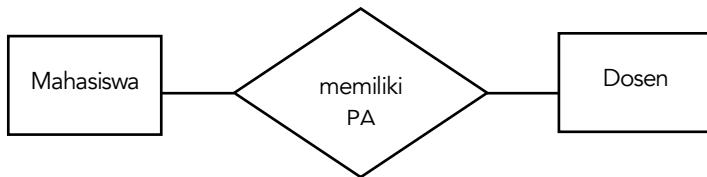
Di dalam penggambarannya, himpunan entitas digambarkan dengan simbol persegi panjang. Pada bagian tengahnya persegi panjang terdapat nama yang dapat merepresentasikan anggota-anggota dari himpunan tersebut. Contoh notasi dari himpunan entitas Mahasiswa dideskripsikan dalam Gambar 4.10.



Gambar 4.7: Penggambaran himpunan entitas Mahasiswa

2. Relasi

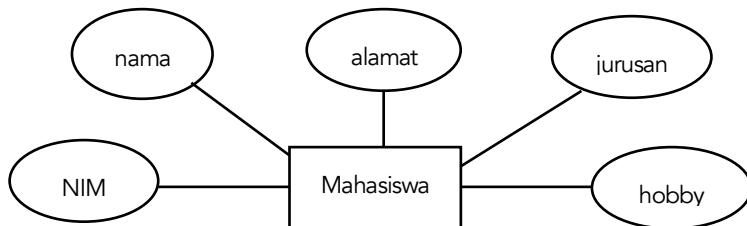
Relasi digunakan untuk menggambarkan adanya hubungan di antara entitas-entitas. Relasi dideskripsikan dengan menggunakan simbol belah ketupat. Di dalamnya terdapat tulisan yang digunakan untuk mendeskripsikan hubungan apa yang terjadi di antara entitas-entitas tersebut. Entitas-entitas yang terlibat dalam suatu relasi dihubungkan dengan menggunakan garis. Contoh dari relasi yang terjadi antara himpunan entitas Mahasiswa dan himpunan entitas Dosen berdasarkan deskripsi dalam Sub Bab 4.1 dan Sub Bab 4.3.2 dideskripsikan dalam Gambar 4.8.



Gambar 4.8: Relasi yang menggambarkan hubungan antara Mahasiswa dengan Dosen

1. Atribut

Atribut merupakan properti dari entitas. Berdasarkan deskripsi pada Sub Bab 4.1, ada lima buah properti yang dimiliki oleh entitas-entitas Mahasiswa, yaitu: NIDN, nama, alamat, jurusan, dan *hobby*. Penggambaran atribut dilakukan dengan menggunakan simbol ellips, di mana di dalamnya terdapat nama dari atribut tersebut. Atribut dihubungkan ke himpunan entitas dengan menggunakan garis. Penggambaran atribut dari himpunan entitas Mahasiswa dideskripsikan dalam Gambar 4.9.



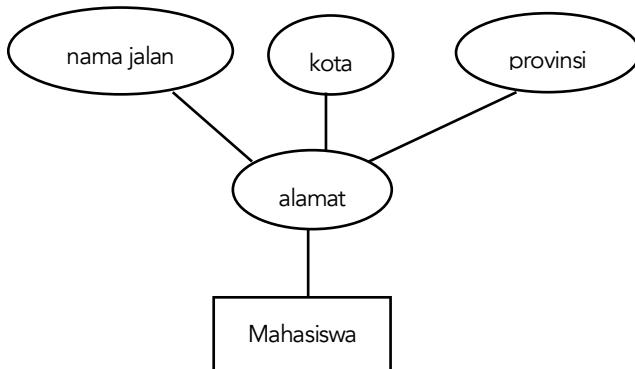
Gambar 4.9: Penggambaran atribut dari himpunan entitas Mahasiswa

Atribut di model data Entity-Relationship memiliki karakteristik sebagai berikut:

a. Atribut sederhana dan atribut komposit

Atribut sederhana adalah atribut yang berisikan nilai tunggal yang tidak dapat dipecah lagi. Contoh dari atribut tunggal berdasarkan deskripsi pada Sub Bab 4.1 adalah NIM, nama (menurut budaya Indonesia yang umumnya tidak dipecah menjadi nama depan, nama tengah, serta nama belakang, kecuali untuk beberapa suku yang menggunakan nama marga), dan jurusan.

Atribut komposit adalah atribut yang dapat dipecah menjadi beberapa atribut. Misalkan alamat yang dapat dipecah menjadi nama jalan, kota, dan provinsi. Atribut komposit dilambangkan dengan ellips yang disusun bertingkat dan dihubungkan dengan garis. Contoh penggunaan atribut komposit dideskripsikan dalam Gambar 4.10.

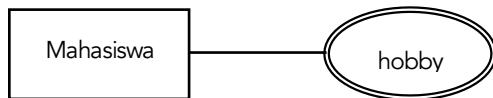


Gambar 4.10: Penggambaran atribut komposit alamat yang disusun dari nama jalan, kota, dan provinsi

b. Atribut *single value* dan atribut *multi value*

Atribut *single value* adalah atribut yang hanya dapat memiliki satu buah nilai. Contoh dari atribut *single value* berdasarkan deskripsi pada Sub Bab 4.1 adalah NIM, nama, jurusan, dan alamat (diasumsikan mahasiswa menuliskan alamat sesuai KTP sehingga satu mahasiswa hanya dapat memiliki satu buah alamat saja).

Atribut *multi value* adalah atribut yang dapat memiliki banyak nilai. Contoh dari atribut *single value* berdasarkan deskripsi pada Sub Bab 4.1 adalah *hobby*. Santi memiliki dua *hobby*, yaitu *travelling* dan membaca novel. Budi juga memiliki dua *hobby* yaitu *travelling* dan bermain futsal. Atribut multi value dideskripsikan dengan menggunakan ellips yang memiliki garis ganda. Contoh penggunaan atribut *multi value* dideskripsikan dalam Gambar 4.11.



Gambar 4.11: Penggambaran atribut komposit alamat yang disusun dari nama jalan, kota, dan provinsi

c. Atribut turunan

Atribut turunan adalah atribut yang nilainya dapat diketahui dari nilai atribut lainnya. Misalnya umur seseorang dapat diketahui dari tanggal lahir. Jika Mahasiswa memiliki atribut tanggal lahir, kita dapat menghitung umur masing-masing mahasiswa. Atribut turunan digambarkan dengan menggunakan ellips di mana garis yang digunakan untuk menggambarkan ellips dibuat putus-putus. Contoh penggunaan atribut *turunan* dideskripsikan dalam Gambar 4.12.



Gambar 4.12: Penggambaran atribut umur yang nilainya dapat diketahui dari atribut *tanggal_lahir*

4.3.4 Model Berorientasi Objek (*Object-Oriented*)

Permasalahan di dunia nyata yang semakin kompleks mengakibatkan timbulnya permintaan model data yang lebih menyerupai dunia nyata. Model data yang lebih menyerupai dunia nyata ini adalah model data berorientasi objek. Pada model data berorientasi objek, data beserta relasinya dikemas dalam suatu struktur data yang disebut dengan objek.

Komponen dari model data berorientasi objek terdiri dari:

- Objek merupakan abstraksi dari sesuatu yang ada di dunia nyata. Objek ini dapat disandingkan dengan entitas pada model data Entity Relationship. Contoh dari objek berdasarkan deskripsi pada Sub Bab 4.1 adalah Jono, Santi, Agus, dan Budi.
- Atribut merupakan properti dari objek. Atribut pada model data berorientasi objek dapat disandingkan dengan atribut pada model data Entity Relationship. Contoh dari atribut pada deskripsi pada Sub Bab 4.1 adalah NIM, nama, jurusan, alamat, dan hobby.
- Class merupakan blueprint dari objek. Objek-objek dicetak berdasarkan class sehingga objek-objek yang berasal dari satu class akan memiliki struktur (atribut) dan *behavior (method)* yang sama.
- Class dapat disusun dalam bentuk hierarki.
- Class dapat mewarisi struktur dan behavior dari class lainnya. Konsep ini disebut dengan *inheritance*.

Database Management System (DBMS) yang menggunakan model berorientasi objek saat ini kurang begitu diminati. Ada dua penyebab kenapa DBMS ini kurang begitu diminati. Pertama, perusahaan umumnya sudah menggunakan basis data berbasis model data relasional. Migrasi dari model Relasional ke model berorientasi objek tentunya membutuhkan biaya yang tidak sedikit bagi perusahaan. Kedua, tidak ada keuntungan yang mendasar dari perubahan model data Relasional ke model data berorientasi objek.

4.3.5 Object/ Relational

Permintaan representasi data yang semakin kompleks mengakibatkan model data Relasional berevolusi, sehingga dapat memenuhi permintaan ini. Evolusi ini menghasilkan suatu model data baru yang disebut dengan *Extended Relational Data Model* (ERDM). Fitur-fitur yang ada di model data berorientasi objek, seperti objek, pengembangan tipe data baru berdasarkan class yang ada, dan *inheritance*, dimasukkan ke dalam basis data relasional. Oleh karena itu *Database Management System* (DBMS) yang didasarkan pada *Extended Relational Data Model* sering disebut sebagai Object/ Relational Database Management System (O/ R DBMS).

Bab 5

Basis Data Relasional

5.1 Pendahuluan

Dahulu permasalahan yang ditemui oleh pengguna ketika mengambil informasi atau melakukan query pada basis data adalah pengguna harus memiliki kemampuan pemrograman tingkat mahir dan mengetahui struktur basis data. Berdasarkan permasalahan tersebut, ahli matematika yang bekerja di IBM San Jose Research Lab bernama Edgar F. Codd mencetuskan konsep model relasional untuk basis data pada tahun 1970. Dalam makalahnya, Edgar F. Codd memberikan gagasan mengenai model relasional di mana informasi yang tersimpan pada basis data yang besar dapat diakses tanpa harus mengetahui bagaimana informasi tersebut disusun dan disimpan pada basis data (IBM, 2020a).

Basis data relasional memberikan independensi data sehingga pengguna dapat mengakses informasi tanpa harus mengetahui detail struktur fisik basis data. Hal tersebut bertujuan membantu pengguna untuk lebih mudah dalam mengakses informasi yang ada pada basis data. Dalam kondisi ini pengguna tidaklah harus ahli dan menguasai pemrograman tingkat mahir. Basis data relasional juga sangat mudah dan sederhana untuk dirancang dan diimplementasikan pada tingkat logis (Gupta & Mittal, 2017). Pengguna menjadi lebih fokus untuk tampilan logis dari basis data daripada memikirkan bagaimana kerumitan data tersebut disimpan pada media penyimpanan.

Basis data relasional menggunakan Structured Query language (SQL) untuk membantu mengelola data dan menjadi mudah untuk dipelajari (Naik, 2014). Pengguna dapat mengelola basis data dengan kemampuan SQL yaitu Data Definition Language (DDL) yang merupakan kumpulan perintah untuk membuat, mengubah dan menghapus basis data dan Data Manipulation Language (DML) untuk melakukan manipulasi data yang terdapat pada basis data. Relational Database Management System (RDBMS) merupakan Database Management System (DBMS) yang menggunakan model relasional dan yang paling populer di industri (Lemahieu, Broucke, & Baesens, 2018). Contoh RDBMS seperti IBM DB2, Oracle, MySQL dan Microsoft SQL Server.

5.2 Definisi Basis Data Relasional

Terdapat beberapa definisi terkait Basis Data Relasional:

1. IBM mendefinisikan basis data relasional sebagai kumpulan data yang diatur ke dalam struktur tabel (IBM, 2020b).
2. Amazon mendefinisikan basis data relasional sebagai kumpulan item data yang hubungannya telah ditentukan sebelumnya dan disusun menjadi satu set tabel dengan kolom dan baris (Amazon, 2020).
3. Oracle mendefinisikan basis data relasional sebagai jenis basis data yang menyimpan dan menyediakan akses ke titik data yang terkait satu sama lain (Oracle, 2020).

Berdasarkan definisi yang telah disampaikan dapat dirangkum pengertian dari basis data relasional adalah kumpulan data yang diatur ke dalam bentuk struktur tabel dan mempunyai relasi yang telah ditentukan sebelumnya.

5.3 Struktur Basis Data Relasional

Rancangan basis data yang baik adalah tujuan dari semua perancangan basis data. Untuk itu terdapat beberapa hal yang harus diketahui oleh perancang basis data relasional untuk membuat basis data yang baik yaitu struktur tabel basis data relasional, kunci dan aturan integritas.

5.3.1 Struktur Tabel Basis Data Relasional

Basis data relasional merepresentasikan data ke dalam bentuk tabel 2 dimensi yang terdiri dari baris dan kolom. Berikut adalah strukur dasar dari tabel basis data relasional:

1. Tabel atau relation.

Pada model basis data relasional tabel disebut juga dengan relation. Tabel terdiri dari kumpulan baris atau record atau tuple. Contoh tabel atau relation dapat dilihat pada Gambar 5.1 yang menunjukkan tabel customer.

Tabel Customer

CustomerID	Name	Email	Status
C001	Anto	anto@contoh.com	Active
C002	Budi	budi@contoh.com	Inactive
C003	Yudhi	yudhi@contoh.com	Active

Gambar 5.1: Contoh Tabel atau Relation

2. Tuple atau Record

Merujuk kepada baris atau record tunggal yang terdapat pada tabel. Setiap tuple terdiri dari kumpulan nilai berdasarkan attribute nya. Pada contoh gambar 5.2 terdapat tiga buah tuple, yaitu tuple pertama {C001, Anto, anto@contoh.com, Active}, tuple kedua {C002, Budi, budi@contoh.com, Inactive}, tuple ketiga {C003, Yudhi, yudhi@contoh.com, Active}.

Tabel Customer

CustomerID	Name	Email	Status
C001	Anto	anto@contoh.com	Active
C002	Budi	budi@contoh.com	Inactive
C003	Yudhi	yudhi@contoh.com	Active

■ Tuple atau Record

Gambar 5.2: Contoh Tuple

3. Attribute

Merujuk kepada kolom yang terdapat pada tabel. Pada contoh gambar 5.3 terdapat empat buah attribute pada tabel Customer yaitu: CustomerID, Name, Email dan Status. Setiap nama attribute harus bersifat unik, tidak boleh ada attribute yang memiliki nama yang sama dalam satu tabel.

Tabel Customer

CustomerID	Name	Email	Status
C001	Anto	anto@contoh.com	Active
C002	Budi	budi@contoh.com	Inactive
C003	Yudhi	yudhi@contoh.com	Active

Attribute

Gambar 5.3: Contoh Attribute

Pada gambar 5.3 juga terdapat domain yang menunjukkan cakupan nilai yang mungkin dari attribute. Pada contoh tersebut sesuai jumlah attribute terdapat empat domain yaitu: Domain CustomerID {C001, C002, C003}, domain Name {Anto, Budi, Yudhi}, domain Email {anto@contoh.com, budi@contoh.com, yudhi@contoh.com} dan domain Status {Active, Inactive, Active}.

4. Degree dan Cardinality

Degree merujuk ke jumlah attribute yang terdapat pada tabel sedangkan cardinality merujuk ke jumlah tuple yang terdapat pada tabel. Pada contoh gambar 5.4, terlihat tabel Customer mempunyai Degree empat dan cardinality tiga. Jumlah degree empat menunjukkan setiap tuple akan mempunyai empat nilai.

Tabel Customer

The diagram shows a single table labeled "Customer". It has four columns: CustomerID, Name, Email, and Status. The table contains three rows with data: C001 (Anto, anto@contoh.com, Active), C002 (Budi, budi@contoh.com, Inactive), and C003 (Yudhi, yudhi@contoh.com, Active). A horizontal double-headed arrow below the table is labeled "Degree". A vertical double-headed arrow to the right of the table is labeled "Cardinality".

CustomerID	Name	Email	Status
C001	Anto	anto@contoh.com	Active
C002	Budi	budi@contoh.com	Inactive
C003	Yudhi	yudhi@contoh.com	Active

Gambar 5.4: Degree dan Cardinality

5.3.2 Kunci

Kunci memiliki peran penting dalam basis data relasional, karena berfungsi untuk memastikan setiap tuple dalam tabel dapat diidentifikasi secara unik, membangun hubungan antara tabel dan memastikan integritas data (Coronel & Morris, 2019). Untuk menggambarkan bagaimana kerja kunci kita akan menggunakan contoh gambar 5.5. Pada gambar tersebut terdapat dua buah tabel yaitu tabel Customer dan tabel Billing.

The diagram illustrates a many-to-many relationship between two tables: "Customer" and "Billing".

Customer Table:

- Primary key:** CustomerID
- Data: C001 (Anto, anto@contoh.com, Active), C002 (Budi, budi@contoh.com, Inactive), C003 (Yudhi, yudhi@contoh.com, Active)

Billing Table:

- Primary Key:** InvoiceNo
- Foreign key:** CustomerID
- Data: B001 (CustomerID C003, Amount 50000), B002 (CustomerID C001, Amount 100000), B003 (CustomerID C003, Amount 30000)

A red vertical line connects the primary key "CustomerID" in the Customer table to the foreign key "CustomerID" in the Billing table.

CustomerID	Name	Email	Status
C001	Anto	anto@contoh.com	Active
C002	Budi	budi@contoh.com	Inactive
C003	Yudhi	yudhi@contoh.com	Active

InvoiceNo	CustomerID	Amount
B001	C003	50000
B002	C001	100000
B003	C003	30000

Gambar 5.5: Relasi Antar Tabel

1. Super Key

Merupakan kunci yang dapat mengidentifikasi tuple pada tabel secara unik. Kunci ini dapat digunakan untuk membedakan setiap tuple yang ada pada tabel. Kunci ini dapat berupa attribute tunggal atau gabungan attribute (composite key). Sebagai contoh Super Key yang memungkinkan dibentuk dari tabel Customer pada gambar 5.5 adalah:

- CustomerID
- Email
- CustomerID+Name
- CustomerID+Email
- CustomerID+Name+Email

2. Candidate Key

Merupakan kunci minimal yang dipilih dari Super Key. Syarat yang harus terpenuhi untuk menjadi Candidate Key adalah tidak adanya data yang redundant. Sebagai contoh untuk Candidate Key yang dapat digunakan pada tabel Customer pada gambar 5.5 adalah:

- CustomerID
- Email

Kedua kunci tersebut dipilih sebagai Candidate Key karena nilai dari attribute tersebut bersifat unik. Setiap Candidate Key merupakan Super Key, tetapi tidak semua Super Key adalah Candidate Key.

3. Primary Key

Merupakan kunci utama yang digunakan untuk mengidentifikasi tuple secara unik pada tabel. Primary key tidak boleh bernilai null. Nilai null merupakan nilai khusus yang menandakan bahwa nilai tersebut tidak diketahui atau tidak ada (Silberschatz, Korth, & Sudarshan, 2020). Primary key harus unik di setiap tuple sehingga tidak ditemukan nilai dari Primary Key yang sama pada tuple lain pada tabel tersebut. Perancang basis data memilih Primary Key dari Candidate key yang telah ditentukan. Sebagai contoh untuk Candidate Key yang dipilih sebagai Primary Key pada tabel Customer pada gambar 5.5 adalah CustomerID.

4. Alternate Key

Merupakan Candidate key yang tidak dipilih sebagai Primary Key. Sebagai contoh untuk Candidate Key yang tidak dipilih sebagai Primary Key pada tabel Customer pada gambar 5.5 adalah Email.

5. Foreign Key (Kunci Tamu)

Merupakan attribute dari suatu tabel yang menunjuk ke Primary Key tabel lainnya sebagai referensi sehingga membentuk relasi atau primary key dari suatu tabel yang menjadi attribute pada tabel lain. Sebagai contoh pada Gambar 5.5, CustomerID yang merupakan Primary Key pada tabel Customer menjadi Foreign Key pada tabel Billing. Relasi yang terjadi dapat dilihat dari kesamaan nama Attribute dari kedua tabel.

5.3.3 Aturan Integritas

Basis data relasional mempunyai aturan integritas untuk memastikan rancangan basis data tetap baik. RDBMS secara otomatis menerapkan aturan integritas ini, tetapi lebih aman untuk memastikan rancangan basis data telah sesuai dengan aturan integritas entitas dan aturan integritas referensial (Coronel & Morris, 2019). Kedua aturan tersebut untuk mengatur Primary Key dan Foreign Key.

1. Aturan Integritas Entitas (Entity Integrity Rules)

Aturan pertama ini memfokuskan kepada Primary key. Aturan pertama ini mewajibkan Primary key bersifat unik (tidak boleh ada nilai yang sama di tuple yang berbeda) dan tidak boleh bernilai null. Tujuan dari aturan ini untuk memastikan Primary Key sebagai pengenal dan pembeda antara tuple yang terdapat pada tabel. Memastikan Primary Key tidak bernilai null membantu Primary Key tereferensi dengan baik ketika menjadi Foreign Key pada tabel lain. Contoh pelanggaran dapat dilihat pada gambar 5.6 dan gambar 5.7. Pada gambar 5.6 dapat dilihat terdapat nilai dari attribute CustomerID yang sama pada dua tuple yaitu C003. Untuk dapat membuatnya baik, perancang basis data harus mengubah salah satu nilai sehingga tidak ada nilai yang sama. Pada gambar 5.7 dapat dilihat terdapat nilai dari CustomerID yang kosong. Untuk dapat membuatnya baik, perancang basi data harus mengisi nilai untuk attribute tersebut dan

memastikan nilai yang dimasukkan tidak memiliki kesamaan dengan nilai dari tuple lainnya.

Tabel Customer
Primary key CustomerID

CustomerID	Name	Email	Status
C001	Anto	anto@contoh.com	Active
C002	Budi	budi@contoh.com	Inactive
C003	Yudhi	yudhi@contoh.com	Active
C004	Rahmat	rahmat@contoh.com	Active
C005	Toni	toni@contoh.com	Active
C003	Tejo	tejo@contoh.com	Inactive

Terdapat nilai yang sama

Gambar 5.6: Pelanggaran Aturan Integritas Entitas, terdapat nilai primary key yang sama

Tabel Customer
Primary key CustomerID

CustomerID	Name	Email	Status
C001	Anto	anto@contoh.com	Active
C002	Budi	budi@contoh.com	Inactive
C003	Yudhi	yudhi@contoh.com	Active
C004	Rahmat	rahmat@contoh.com	Active
	Toni	toni@contoh.com	Active
C006	Tejo	tejo@contoh.com	Inactive

Terdapat nilai yang null

Gambar 5.7: Pelanggaran Aturan Integritas Entitas, terdapat nilai primary key yang null

2. Aturan Integritas Referensial (Referential Integrity Rules)

Aturan kedua memfokuskan kepada Foreign Key. Di mana nilai foreign key dapat saja null tetapi nilai tersebut harus sesuai dengan nilai yang

terdapat pada attribute Primary Key yang terdapat pada tabel yang direferensikan. Pada gambar 5.8 terdapat pelanggaran yaitu pada tabel Billing pada attribute CustomerID yaitu C006. Pelanggaran terjadi karena tidak terdapat nilai C006 pada tabel Customer. Di mana kita ketahui CustomerID menjadi kunci tamu pada tabel Billing sehingga nilai yang terdapat pada attribute CustomerID di tabel Billing harus sesuai dengan nilai yang ada pada tabel Customer.

Tabel Customer
Primary key CustomerID

CustomerID	Name	Email	Status
C001	Anto	anto@contoh.com	Active
C002	Budi	budi@contoh.com	Inactive
C003	Yudhi	yudhi@contoh.com	Active

Tabel Billing
Primary Key InvoiceNo
Foreign key CustomerID

InvoiceNo	CustomerID	Amount
B001	C003	50000
B002	C001	100000
B003	C003	30000
B004	C006	50000
B005	C003	30000

Gambar 5.8: Pelanggaran Aturan Integritas Referensial, terdapat nilai yang tidak sesuai

5.4 Hubungan Dalam Basis Data Relasional

Hubungan pada basis data relasional dibagi menjadi tiga yaitu one-to-one (1:1), one to many (1:M) dan many to many (M:N). Hubungan one to one (1:1) menunjukkan setiap nilai attribute yang digunakan sebagai Foreign Key hanya untuk satu tuple. Sebagai contoh gambar 5.9 yang menunjukkan hubungan one to one yang terjadi pada antara tabel Country dengan tabel Capital. Dapat dilihat

nilai Attribute CountryID pada tabel Country yang menjadi Foreign Key pada tabel Capital hanya digunakan di satu tuple.

Tabel Country
Primary key CountryID

CountryID	Name
1	Amerika serikat
62	Indonesia
81	Jepang

Tabel Capital
Primary Key CapitalID
Foreign key CountryID

CapitalID	CountryID	Name
US	1	Washington, D.C
ID	62	Jakarta
JP	81	Tokyo

Gambar 5.9: Hubungan one to one

Hubungan one to many (1:M) menunjukkan setiap nilai attribute dapat digunakan sebagai Foreign Key pada banyak tuple. Pada gambar 5.10 nilai attribute pada tabel Customer yaitu CustomerID digunakan pada beberapa tuple yang ada pada tabel Billing. Dapat dilihat pada nilai attribut CustomerID pada contoh yaitu C003.

Tabel Customer
Primary key CustomerID

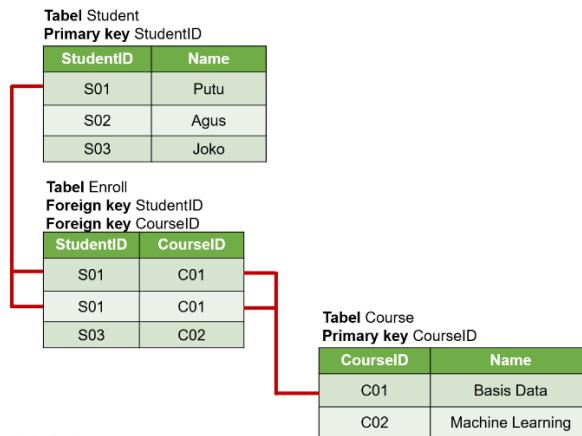
CustomerID	Name	Email	Status
C001	Anto	anto@contoh.com	Active
C002	Budi	budi@contoh.com	Inactive
C003	Yudhi	yudhi@contoh.com	Active

Tabel Billing
Primary Key InvoiceNo
Foreign key CustomerID

InvoiceNo	CustomerID	Amount
B001	C003	50000
B002	C001	100000
B003	C003	30000

Gambar 5.10: Hubungan one to many

Hubungan Many to many (M:N) tidak dapat dilakukan secara langsung pada basis data relasional. Sebagai solusi dari permasalahan tersebut dibuatlah tabel yang berfungsi untuk penghubung. Pada gambar 5.11, Hubungan antara tabel Student dengan tabel course adalah many to many. Di mana satu siswa (student) dapat mengambil banyak kursus (Course), begitu pula sebaliknya satu kursus bisa diambil oleh banyak siswa. Sehingga dibuatlah tabel Enroll yang menampung kondisi tersebut. Hubungan antara tabel student dengan tabel Enroll menjadi one to many dan hubungan antara tabel course dengan tabel Enroll juga menjadi one to many.



Gambar 5.11: Hubungan Many to Many

Basis data relasional merupakan jenis basis data yang merepresentasikan data ke dalam bentuk tabel 2 dimensi yang terdiri dari baris dan kolom dan mempunyai relasi yang telah ditentukan sebelumnya. Struktur tabel basis data relasional terdiri dari tabel (relation), tuple (baris), attribute (kolom), *degree* dan *cardinality*. Kunci digunakan untuk memastikan setiap tuple teridentifikasi dengan unik, memastikan hubungan antar tabel dan integritas data. Terdapat beberapa kunci pada basis data relasional yaitu *super key*, *candidate key*, *primary key*, *alternate key* dan *foreign key*. Untuk memastikan rancangan basis data yang baik, basis data relasional mempunyai dua aturan integritas yaitu aturan integritas entitas yang berfokus untuk Primary Key dan aturan integritas referensial yang berfokus pada Foreign Key. Hubungan antara tabel pada basis data relasional dibagi menjadi tiga yaitu *one to one*, *one to many* dan *many to many*.

Bab 6

Entitas, Atribut dan Relasi

6.1 Entitas

Entitas merupakan sekumpulan obyek yang saling terkait dan keterangannya ada didalam sistem. Jika dikaitkan dengan dunia nyata, maka entitas ini dapat dicontohkan dengan subyek atau orang, benda dan hal lainnya. Pandangan lainnya oleh (Harrington, 2016) menyebutkan bahwa sebuah entitas adalah sebuah “benda” (*thing*) atau “objek” (*object*) di dunia nyata dan secara spesifik bisa dibedakan dengan semua objek lainnya. Tidak semua benda dan objek bisa disebut entitas, karena harus dapat dibedakan dan diidentifikasi secara unik dengan objek lainnya, sehingga semua contoh tersebut bisa dikatakan sebuah entitas adalah ketika keterangannya perlu disimpan didalam basis data.

Dalam proses menggambarkan entitas terdapat beberapa aturan yaitu:

1. Simbol entitas dinyatakan dengan persegi panjang
2. Penamaan entitas dituliskan di dalam simbol persegi panjang
3. Penamaan entitas bisa menggunakan kata benda dan tunggal
4. Nama entitas menggunakan penamaan yang mudah dimengerti serta memiliki arti yang jelas.
5. Nama entitas diharapkan singkat dan spesifik sehingga tidak menyulitkan pemakai.

Seringkali nama entitas dapat tersusun atas lebih dari satu kata. Untuk memenuhi aturan penggambaran tersebut diatas, maka sering digunakan tanda (garis bawah/ *under score*) yang dimaksud untuk menyatakan bahwa beberapa kata tersebut dianggap kata tunggal.

Misalnya :

1. Untuk menyatakan nama entitas retur pembelian akan lebih baik dinamakan **Retur_Pembelian** bukan **Retur** atau **Pembelian**, karena lebih mudah dipahami dan tidak menimbulkan interpretasi yang berbeda bagi pemakai.
2. Untuk menyatakan nama entitas pengeluaran barang akan lebih baik dinamakan sebagai **Pengeluaran_Barat** bukan **Pengeluaran** atau **Barat**, karena lebih mudah dipahami dan tidak menimbulkan interpretasi yang berbeda bagi pemakai.

Dalam kasus yang lain, seringkali nama entitas yang panjang justru menyulitkan, pemakai akan lebih memahami dengan nama yang lebih singkat. Misalnya: untuk menyatakan nama entitas retur total pembelian barang bisa dinamakan dengan **Retur_Pembelian**. Penggunaan singkatan bisa digunakan dalam penamaan entitas selama arti dari entitas sudah cukup jelas untuk dipahami. Penentuan entitas dalam suatu sistem perlu dilakukan dengan cermat dan hati-hati. Tidak semua orang, benda atau hal dapat disebut entitas. Hanya orang, benda, dan hal yang terkait dengan sistem dan keterangannya perlu disimpan dalam basis data saja yang dapat disebut entitas.

Sebagai contoh, dalam suatu subsistem pengolahan data penjualan, entitas yang terlibat dalam subsistem tersebut dapat meliputi orang seperti ditunjukkan pada Gambar 6.1, entitas berupa benda seperti ditunjukkan oleh Gambar 6.2, dan entitas berupa hal-hal seperti ditunjukkan oleh Gambar 6.3.

Objek Dasar	Simbol Entitas
Pelanggan	Pelanggan
Pemasok	Pemasok
Karyawan	Karyawan

Gambar 6.1 Contoh-contoh entitas berupa orang

Objek Dasar	Simbol Entitas
Barang	Barang

Gambar 6.2 Contoh-contoh entitas berupa benda

Objek Dasar	Simbol Entitas
Pembelian	Pembelian
Penjualan	Penjualan
Retur Pembelian	Retur_Pembelian
Retur Penjualan	Retur_Penjualan
Pemesanan Penjualan	Pemesanan_Penjualan
Pembayaran Pembelian	Pembayaran_Pembelian

Gambar 6.3 Contoh-contoh entitas berupa hal

6.1.1 Isian Entitas

Isian entitas adalah sebuah atau sekumpulan baris data (*record*) yang ada pada entitas. Contoh isian entitas :

- Barang dengan Kode_Barang, contohnya: BRG-010
- Barang dengan Nama_Barang, contohnya: Sepatu
- Pembelian dengan Kode_Pembelian, contohnya : SP001
- Retur Penjualan dengan Kode_Retur_Penjualan, contohnya: R002

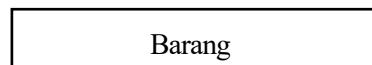
6.1.2 Himpunan Entitas

Himpunan entitas atau sering disebut *entity sets* adalah sekumpulan entitas yang seringkali memiliki property, struktur/sifat yang sama(Silberschatz, Korth and Sudarshan, 2011). Contoh himpunan entitas yaitu sejumlah barang.

6.1.3 Jenis Entitas

Entitas dapat dibedakan berdasarkan jenisnya yaitu entitas lemah dan entitas kuat. Dilihat dari sifatnya, entitas ini akan sangat berpengaruh didalam proses pembuatan ER_M (*Entity Relationship Model*). Secara definisi jenis entitas dapat dijelaskan sebagai berikut:

1. Entitas regular atau kuat disebut juga entitas dominan yaitu entitas yang kemunculannya tidak bergantung dengan entitas lainnya. Jika dikaitkan dengan ER_M maka entitas kuat ini menjadi entitas yang selalu menjadi entitas penentu dalam pembuatan entitas lain. Misalnya : entitas Barang, entitas Pelanggan, entitas Pemasok. simbol untuk entitas kuat adalah persegi Panjang.



Gambar 6.4. Entitas Barang

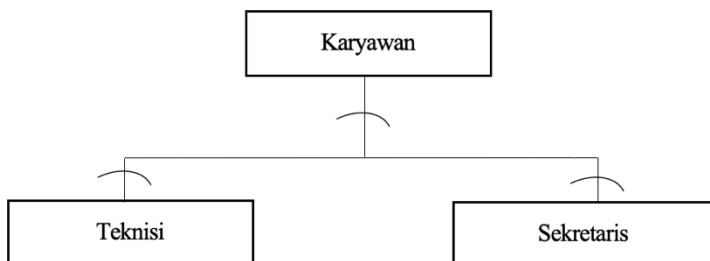
2. Entitas dependen atau lemah adalah entitas yang sangat bergantung dengan entitas lainnya, atau bisa dijelaskan kemunculannya sangat bergantung dengan entitas kuat. Simbol entitas lemah yaitu dua persegi Panjang (dobel). Contohnya: entitas Barang_Masuk sangat bergantung dengan Entitas Barang.



Gambar 6.5. Entitas Barang_Masuk

3. Entitas *super type* dan *sub type*

Sebuah entitas bisa jadi mempunyai hubungan antar entitas dengan sifat bahwa salah satu entitas menjadi bagian dari entitas yang lainnya. Entitas *super type* adalah entitas dengan tingkatan lebih tinggi dari entitas *sub type*, entitas ini mempunyai entitas dengan tingkatan dibawahnya (Teorey *et al.*, 2011). Sedangkan entitas *sub type* adalah entitas dengan tingkatan lebih rendah, yaitu entitas yang menjadi bagian entitas lain yaitu entitas *super type*. Sebagai contoh, entitas teknisi dan sekretaris merupakan bagian dari entitas Karyawan. Entitas Karyawan disebut sebagai entitas *super type* (*super type entity*), sedangkan entitas teknisi dan sekretaris disebut sebagai entitas *sub type* (*sub type entity*), terhadap entitas Karyawan. Untuk menyatakan entitas *super type* dan *sub type*, ditunjukkan oleh Gambar 6.6.



Gambar 6.6 Entitas *super type* dan *sub type*

6.2 Atribut

Suatu entitas diwakili oleh sekumpulan atribut, sehingga atribut adalah *property* yang mendeksripsikan setiap himpunan entitas. Atribut berisikan keterangan-keterangan yang terkait dengan sebuah entitas dan menyimpan informasi serupa dalam basis data (Korth and Silberschatz, 2010). Namun setiap entitas memiliki nilai atribut yang berbeda-beda. Di antara atribut tersebut harus terdapat atribut unik yang dapat membedakan atribut dari setiap kumpulan entitas. Contohnya adalah atribut Kode_barang pada entitas barang. Atribut Kode_Barang digunakan untuk mengidentifikasi setiap barang secara unik., karena mungkin ada lebih dari satu barang yang disimpan dengan nama yang sama. Dalam proses menggambarkan entitas terdapat beberapa aturan yaitu:

1. Penamaan atribut dituliskan di dalam simbol ellips
2. Penamaan atribut bisa menggunakan kata benda dan tunggal
3. Nama atribut menggunakan penamaan yang mudah dimengerti dan jelas.
4. Atribut dan entitas yang saling berelasi dihubungkan menggunakan simbol garis.

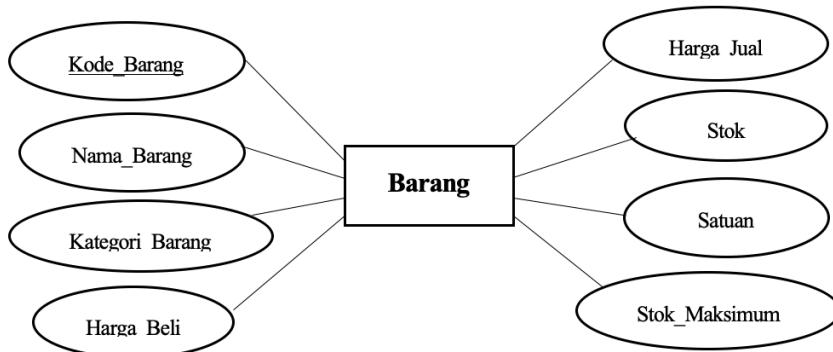
Penamaan atribut yang lebih dari dua kata jika diperlukan menggunakan tanda _ (garis bawah/*underscore/hyphen*), atau penggunaan singkatan diperbolehkan selama lebih mudah dipahami. Contoh-contoh atribut untuk setiap entitas dalam contoh subsistem pengolahan data perjualan dengan atribut unik setiap entitas yang ditulis tebal dan digarisbawahi, dapat ditampilkan pada Tabel 6.1.

Tabel 6.1: Contoh-Contoh Atribut

Simbol Entitas	Atribut
Barang	Kode Barang , Nama_Barang, Kategori_Barang, Harga_Beli, Harga_Jual, Stok, Satuan, Stok_Maksimum
Pelanggan	Kode Pelanggan , Nama_Pelanggan, Alamat_Pelanggan, Kota, Kode_Pos, Propinsi, Telp_Pelanggan
Pemasok	Kode Pemasok , Nama_Pemasok, Alamat_Pemasok, Kota, Kode_Pos, Propinsi, Telp_Pemasok
Karyawan	Kode Karyawan , Nama_Karyawan, Jenis_Kelamin, Alamat_Karyawan, Kota, Kode_Pos, Propinsi, Telp_Karyawan, Jabatan
Pembelian	Kode Pembelian , Tgl_Pembelian,ode_Pemesanan_Pembelian, Qty_Pembelian, Harga_Pembelian, Jenis_Pembayaran
Penjualan	Kode Penjualan , Tgl_Penjualan, Kode_Pemesanan_Penjualan, Qty_Penjualan, Harga_Penjualan, Jenis_Pembayaran

Retur Pembelian	<u>Kode Retur Pembelian</u> , Tgl Retur_Pembelian, Qty_Retur_Pembelian, Alasan_Retur
Retur Penjualan	<u>Kode Retur Penjualan</u> , Tgl Retur_Penjualan, Qty_Retur_Penjualan, Alasan_Retur

Sebagai contoh, penggambaran atribut pada entitas Barang menggunakan aturan Chen Model (Coronel and Morris, 2016) ditunjukkan oleh Gambar 6.7.



Gambar 6.7 Contoh atribut pada entitas Barang

6.2.1 Pengelompokan Atribut

Atribut pada sebuah entitas dapat diklasifikasikan dalam beberapa kelompok (Indrajani, 2017) (Kusrini, 2007), yaitu :

1. Atribut sederhana (*simple attribute*), yaitu jika atribut berisi sebuah komponen nilai/elementer dan tidak bisa dibagi lagi kedalam komponen yang lebih kecil atau dikenal dengan atribut atomic. Contoh atribut sederhana dalam entitas Barang yang ditunjukkan pada Tabel 6.1 adalah Kode Barang.
2. Atribut komposit (*composite attribute*), yaitu atribut yang komponen atau bagianya terdapat melebihi satu nilai pada setiap proses. Contoh atribut komposit dan nilai atribut dalam entitas Karyawan yang ditunjukkan pada Tabel 6.1 adalah atribut Alamat_Karyawan yang bisa terdiri dari jalan, kota, dusun, kecamatan, kabupaten dan lain-lain.
3. Atribut tunggal (*single-valued attribute*) yaitu jika atribut mempunyai nilai yang tunggal pada setiap proses. Contoh atribut bernilai tunggal

adalah dalam entitas Karyawan yang ditunjukkan pada Tabel 6.1 adalah atribut Jenis_Kelamin.

4. Atribut bernilai banyak (*multi-valued attribute*) yaitu jika atribut mempunyai beberapa nilai pada sebuah kejadian. Contoh atribut bernilai bernilai banyak dalam entitas Karyawan yang ditunjukkan pada Tabel 6.1 adalah No_Telp.
5. Atribut Turunan (*derived attribute*) yaitu jika atribut dihasilkan dari atribut lainnya atau atribut turunan. Contoh atribut turunan dalam entitas penjualan yang ditunjukkan pada Tabel 6.1 adalah atribut harga_jual.
6. Atribut kunci (*Key*) yaitu atribut yang dipergunakan untuk menentukan nilai unik pada setiap kumpulan entitas. Atribut kunci terdiri dari sekumpulan atribut unik yang dapat membuat atribut menjadi spesifik pada suatu entitas(Hutahaean, 2015). Atribut kunci di antaranya:
 - a. *Super Key* merupakan kumpulan atribut unik yang mampu membuat setiap baris data (*record*) terlihat berbeda pada tabel. Contoh superkey dalam entitas Pelanggan dapat dilihat pada Tabel 6.1 adalah atribut Kode_Pelanggan.
 - b. *Candidate Key* merupakan sekumpulan minimal dari kumpulan atribut yang membuat setiap baris data berbeda atau unik dalam tabel. Untuk bisa menjadi kandidat key maka atribut harus memenuhi syarat sebagai sebuah superkey dengan jumlah atributnya paling sedikit. Contoh kandidat key dalam entitas Barang dapat dilihat pada Tabel 6.1 adalah atribut Kode_Barang.
 - c. *Primary Key* atau sering disebut kunci relasi/kunci utama terdiri dari beberapa atribut. Untuk menjadi primary key maka atribut tersebut adalah kandidat key yang memiliki fungsi untuk mengidentifikasi dan membedakan baris data secara unik. Contoh primary key dalam entitas Barang yang ditunjukkan pada Tabel 6.1 adalah atribut Kode_Barang.
 - d. *Composite Key* merupakan suatu primary key yang memenuhi persyaratan yaitu memiliki lebih dari 1 atribut.
 - e. *Foreign Key* merupakan sekumpulan atribut dalam suatu relasi yang kumpulan atribut ini adalah *primary key* dari atribut dalam entitas lain dalam sebuah relasi.

- f. *Alternate Key* merupakan suatu atribut yang dibuat menjadi kunci alternatif apabila tidak ada satupun atribut yang bisa dijadikan kandidat key dalam sebuah relasi.

6.3 Relasi Antar Entitas

Relasi atau kerelasian antar entitas mendefinisikan hubungan antar entitas dimana hubungan ini mendeskripsikan kejadian atau proses yang saling terkait di antara dua entitas (Silberschatz, Korth and Sudarshan, 1997), serta keterangannya disimpan ke basis data.

Dalam proses menggambarkan relasi terdapat beberapa aturan yaitu :

1. Relasi dinyatakan dengan simbol belah ketupat.
2. Penamaan relasi dituliskan dalam simbol belah ketupat.
3. Penamaan relasi menggunakan kata kerja aktif (diawali dengan awalan me) dan tunggal.
4. Nama relasi menggunakan penamaan yang mudah dimengerti dan jelas.

Relasi antar entitas ini juga disebut dengan derajat kardinalitas relasi yang menyatakan jumlah entitas yang dapat dikaitkan dengan entitas lainnya melalui sebuah hubungan atau relasi. Derajat kardinalitas relasi sangat efektif digunakan untuk mendeskripsikan hubungan antar entitas. Derajat kardinalitas relasi antar entitas dapat dikelompokkan dalam tiga jenis, yaitu :

1. Kerelasian jenis 1-ke-1/ satu ke satu (*one to one*)

Relasi yang terjadi antar entitas jika satu baris pada entitas pertama dihubungkan hanya satu baris data pada entitas ke dua atau jika nilai yang digunakan sebagai penghubung pada entitas pertama hanya memungkinkan muncul satu kali saja pada entitas kedua yang saling berhubungan.

Sebagai contoh, **satu** orang pelanggan (atribut-atributnya tersimpan dalam entitas Pelanggan) hanya dimungkinkan **melakukan** **satu** **retur_pembelian** (atribut-atributnya tersimpan dalam entitas Retur_Pembelian).

2. Kerelasian jenis n-ke-1/ banyak ke satu (*many to one*) atau 1-ke-n/ satu ke banyak (*one to many*)

Relasi yang terjadi antar entitas jika satu baris pada entitas pertama dihubungkan dengan lebih dari satu baris data pada entitas ke dua atau jika nilai yang digunakan sebagai penghubung pada entitas pertama hanya memungkinkan muncul lebih dari satu kali saja pada entitas kedua yang saling berhubungan.

Sebagai contoh, **lebih dari satu** pelanggan (atribut-atributnya tersimpan dalam entitas Pelanggan) dapat **membeli** hanya **satu** barang (atribut-atributnya tersimpan dalam entitas Barang). Kondisi seperti ini disebut jenis kerelasian **n-ke-1**.

Tetapi sebaliknya, satu buah barang (atribut-atributnya tersimpan dalam entitas Barang) dapat **dibeli** oleh **lebih dari satu** pelanggan (atribut-atributnya tersimpan dalam entitas Pelanggan). Kondisi seperti ini disebut jenis kerelasian 1-ke-n.

3. Kerelasian jenis n-ke-n/banyak ke banyak (*many to many*)

Relasi yang terjadi antar entitas jika banyak baris pada entitas pertama dihubungkan dengan lebih dari satu baris data pada entitas ke dua dan sebaliknya. Jika nilai atribut penghubung pada entitas pertama dimungkinkan muncul lebih dari satu proses, baik pada entitas pertama maupun entitas kedua yang saling berhubungan.

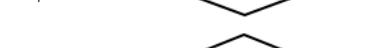
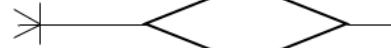
Sebagai contoh, **lebih dari satu** karyawan (atribut-atributnya tersimpan dalam entitas Karyawan) dapat melakukan lebih **dari satu** proses penjualan (atribut-atributnya tersimpan dalam entitas Penjualan). Kondisi seperti ini disebut jenis kerelasian **n-ke-n**.

Suatu kerelasian antara dua buah entitas akan merepresentasikan suatu kunci penghubung. Jenis kerelasian 1-ke-1 dan n-ke-n, umumnya memerlukan file transaksi, sedangkan jenis kerelasian 1-ke-n umumnya memerlukan atribut sebagai kunci penghubung. Jenis kerelasian sangat bergantung pada aturan bisnis (*business rule*) yang digunakan pada sistem/organisasi.

6.3.1 Simbol Relasi Antar Entitas

Dalam kaitannya dengan perancangan ER_M maka simbol relasi sangat penting digunakan agar memudahkan pemakai dalam merancang relasi antar entitas sesuai dengan persoalan masing-masing. Terdapat beberapa jenis simbol relasi antar entitas dalam hal penggunaan simbol kerelasian antar entitas dan untuk menggambarkan kerelasian (Teorey *et al.*, 2011).

1. Pilihan ke-1

Jenis Kerelasian	Simbol yang digunakan
1-ke-1	: 
1-ke-n	: 
n-ke-1	: 
n-ke-n	: 

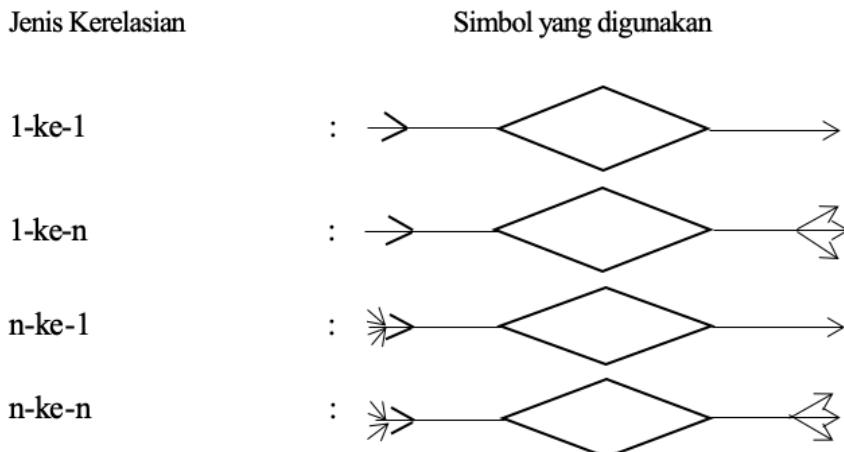
Gambar 6.8: Simbol kerelasian antar entitas (pilihan 1)

2. Pilihan ke-2

Jenis Kerelasian	Simbol yang digunakan
1-ke-1	:
1-ke-n	:
n-ke-1	:
n-ke-n	:

Gambar 6.9: Simbol kerelasian antar entitas (pilihan 2)

3. Pilihan ke-3



Gambar 6.10: Simbol kerelasian antar entitas (pilihan 3)

Saat menggambarkan ER_D, kita dapat menggunakan salah satu dari pilihan diatas sesuai keinginan kita, yang penting penggunaan simbol tersebut dilakukan secara konsisten. Artinya, kalau kita menggunakan simbol kelompok pilihan 1, maka untuk seluruh bagian dalam ER_D harus menggunakan simbol dalam pilihan 1 tersebut. Jika menggunakan simbol dalam pilihan 2, maka semua bagian ER_D harus menggunakan simbol dalam pilihan 2. Demikian juga jika yang digunakan adalah pilihan 3, maka seluruh bagian ER_D harus menggunakan simbol pilihan 3. Sebagai contoh untuk menunjukkan hubungan antar entitas Karyawan dan entitas Penjualan dengan nama kerelasian mengikuti, dengan jenis n-ke-n, maka dapat digambarkan sebagaimana ditunjukkan oleh Gambar 6.11 (dalam pilihan 1), Gambar 6.12 (dalam pilihan 2) dan Gambar 6.13 (dalam pilihan 3)



Gambar 6.11: Contoh kerelasian antara entitas Karyawan dan Penjualan (pilihan 1)



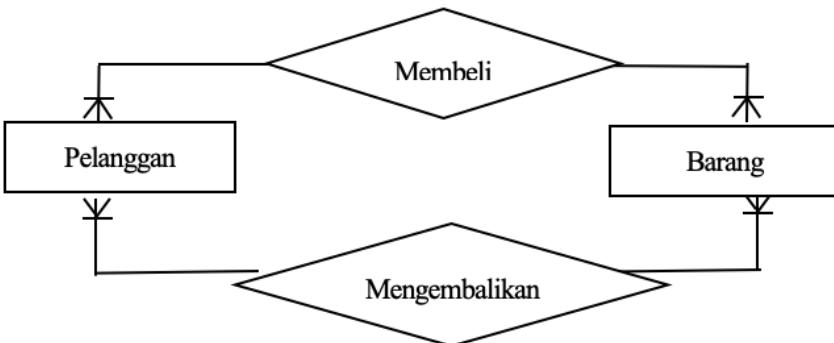
Gambar 6.12: Contoh kerelasian antara entitas Karyawan dan Penjualan (pilihan 2)



Gambar 6.13: Contoh kerelasian antara entitas Karyawan dan Penjualan (pilihan 3)

6.3.2 Relasi Berganda

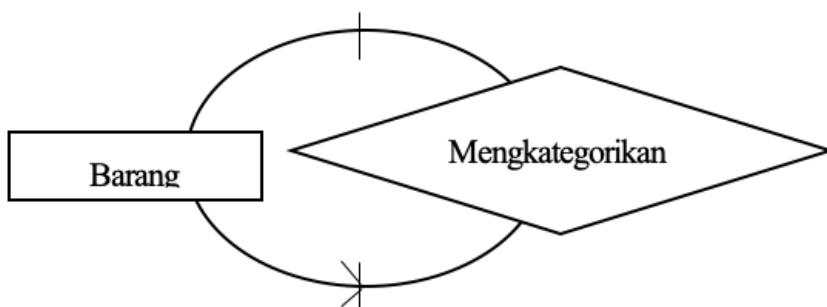
Dalam kerelasian antara dua buah entitas, dimungkinkan terjadinya dua kerelasian sekaligus di antara dua entitas tersebut. Kondisi relasi ini disebut sebagai kerelasian berganda. Contoh relasi berganda adalah kerelasian antara entitas Pelanggan dan Barang. Pelanggan dapat membeli Barang di suatu toko, dan Pelanggan juga bisa mengembalikan Barang ke toko, dengan jenis kerelasian 1-ke-n. Relasi berganda antara entitas Pelanggan dan Barang tersebut dapat digambarkan sebagaimana ditunjukkan oleh Gambar 6.14.



Gambar 6.14 Contoh Relasi Berganda

6.3.3 Relasi Rekursif

Relasi rekursif terjadi apabila entitas memiliki relasi dengan entitas yang sama. Sebagai contoh, dalam entitas Barang, bisa jadi sebuah barang dapat dikategorikan sebagai Barang_Masuk dan Barang_Keluar, kedua jenis barang ini dapat dikategorikan apabila entitas barang sudah tercatat dalam basis data. Kondisi ini disebut sebagai kerelasian rekursif, dengan jenis kerelasian 1-ke-n. Kerelasian rekursif entitas Barang tersebut digambarkan sebagaimana ditunjukkan oleh Gambar 6.15.



Gambar 6.15 Contoh kerelasian rekursif

6.3.4 Relasi Asosiatif

Kerelasian asosiasif, terjadi jika kerelasian di antara dua buah entitas yang berbeda dan dari relasi itu menghasilkan beberapa informasi. Sebagai contoh, hubungan antara entitas Pemasok dan Pemesanan_Barang dapat menunjukkan kerelasian asosiasif dengan menghasilkan informasi seperti Tanggal_Pemesanan, Harga_Barang dan atau diskon_barang. Kondisi ini disebut sebagai kerelasian asosiasif, dengan jenis kerelasian 1-ke-n. Kerelasian asosiasif antara entitas Pemasok dan Pemesanan_Barang tersebut dapat digambarkan sebagaimana ditunjukkan oleh Gambar 6.16.



Gambar 6.16 Contoh kerelasian Asosiasif

Bab 7

Entity Relationship Diagram

7.1 Pendahuluan

Pada pengembangan sebuah perangkat lunak, terdapat salah satu tahapan merancang sistem basis data didalamnya. Di mana proses perancangan basis data didapatkan dari perancangan model konseptual dan relasional. Proses yang cukup sulit dilakukan adalah pada saat pemetaan konsep basis data, apabila kebutuhan pengguna mengalami perubahan. Di mana artinya perubahan tersebut berdampak pada perancangan basis data sebelumnya, selain itu struktur basis data yang kompleks akan mempengaruhi waktu pengerjaan dari pemetaan konsep basis data. (Larassati *et al.*, 2019). Untuk menggambarkan konsep basis data tersebut dikenal dengan istilah Diagram Relasi Entitas atau biasa dikenal dengan istilah *Entity Relationship Diagram* (ERD). ERD menggambarkan entity-relationship model yang merupakan gabungan konsep entitas, atribut, dan hubungan antar entitas, dan entitas dalam ERD merepresentasikan sesuatu (*things*) atau benda dalam dunia nyata. ERD merupakan diagram yang menggambarkan relationship antar entitas yang relevan dari system interest. Entitas dapat dijelaskan sebagai sesuatu yang menyebabkan seseorang mengumpulkan data untuk diperoses menjadi informasi pendukung kegiatan bisnis. Pada aplikasi IT traditional view driven, konsep model data hanya berfokus pada pemilihan struktur data dari satu atau beberapa cara pandang. Dalam model data “event-driven” digunakan notasi

yang sama dengan model data “viewdriven”, tetapi fokus dan sasaran modeling data berbeda. Model data event-driven fokus pada kegiatan bisnis untuk seluruh proses bisnis, bukan hanya melihat fungsi-fungsi organisasi.(Adi and Kristin, 2014). Dengan kata lain pembentukan ERD didasarkan pada kebutuhan pengguna, di mana setiap objek yang terlibat dalam penggunaan perangkat lunak tersebut diwakili oleh entitas dengan properti komponennya diwakili oleh atributnya. Terdapat tiga komponen atau notasi utama pada pembentukan sebuah ERD yaitu entitas, atribut dan relasinya. Dalam pembuatan sebuah ERD terdapat beberapa tahapan, tetapi sebelumnya harus dipatikan terlebih dahulu konsep dari membangun sebuah ERD.

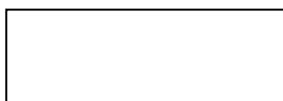
7.2 Konsep Entity Relationship Diagram

Untuk membuat sebuah Entity Relationship Diagram atau ERD terdapat tiga komponen atau notasi utama yaitu entitas, atribut dan relasi. Sebuah entitas adalah sebuah benda (things) atau objek yang didunia nyata yang dapat dibedakan dari semua objek lainnya. Kumpulan entitas dengan dengan jenis yang sama disebut entitas set. Di mana kesamaannya dapat dilihat dari kesamaan atribut dan propertinya pada setiap entitinya (Kusrini,2007).

Agar lebih mudah memahami definisi tentang entitas set, dapat dilihat dari contoh berikut :

1. Kumpulan orang yang bekerja pada sebuah perusahaan dapat disebut sebagai entiti set pegawai.
2. Kumpulan orang yang diperiksa dalam sebuah rumah sakit dapat disebut sebagai entiti set pasien.

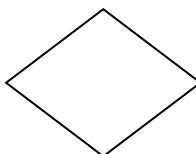
Entiti set dapat disimbolkan dalam bentuk persegi panjang yang dapat dilihat pada gambar 7.1.



Gambar 7.1: Simbol Entiti set

Komponen kedua adalah relasi, yaitu hubungan yang terjadi antara dua buah entitas atau lebih. Kumpulan relasi pada ERD disebut relasi set. Relasi set

dapat disimbolkan dalam bentuk belah ketupat yang dapat dilihat pada gambar 7.2 berikut.



Gambar 7.2: Simbol Relasi set

Komponen yang ketiga adalah atribut, di mana atribut merupakan berisi data properti dari setiap entitas. Atribut pada ERD disimbolkan dalam bentuk oval, seperti pada gambar 7.3.



Gambar 7.3: Simbol atribut

Pada pembuatan ERD terdapat beberapa istilah lain yang digunakan, antara lain : (Kusrini, 2007)

1. Superkey

Superkey adalah satu atau lebih atribut (kumpulan atribut) yang dapat membedakan setiap baris data dalam tabel secara unik. Sebagai contoh diberikan tabel pegawai sebagai ilustrasi :

Tabel 7.1: Tabel Pegawai

NIP	Nama	Alamat	Golongan
2020.01.016	Amir	Denpasar	III B
2020.01.023	Santi	Gianyar	III C
2020.02.017	Agus	Tabanan	III B

Pada tabel 7.1 yang mungkin menjadi superkey adalah atribut NIP. Atribut NIP dan Nama, atribut NIP dan Alamat, atribut NIP dan Golongan, atribut NIP, Nama dan Alamat, atribut NIP, Nama dan Golongan.

Atribut Nama tidak bisa dijadikan superkey karena kemungkinan terdapat lebih dari satu baris memiliki nilai data yang sama.

2. Kandidat key

Kandidatkey adalah atribut minimal yang membedakan setiap baris data dalam tabel secara unik. Berarti kandidat key adalah superkey yang paling sedikit jumlah atributnya. Sehingga dari tabel 7.1 yang memenuhi kriteria menjadi kandidat key adalah atribut NIP.

3. Kunci Relasi (*Relation Key*)/Kunci Utama (*Primary Key*)

Nilai dari kunci relasi harus mengidentifikasi sebuah baris yang unik di dalam sebuah relasi. Agar bisa menjadi kunci utama sebuah atribut haruslah memenuhi persyaratan sebagai kandidat key. Sebagai contoh pada tabel 7.1, pada tabel pegawai yang menjadi kunci relasinya adalah NIP.

4. Kunci Alternatif (Alternate key)

Kunci yang tidak ada di dunia nyata tetapi dibuat menjadi ada dan dijadikan primary key. Kunci alternatif dibuat ketika tidak ada sama sekali atribut pada sebuah relasi yang dapat digunakan atau tidak efektif apabila digunakan sebagai primary key.

Sebagai contoh tabel 7.2 yaitu tabel Keahlian. Pada tabel Keahlian yang menjadi *kandidat key* adalah Nama_keahlian, untuk mempermudah biasanya dibuatkan satu kunci alternatif untuk menandai bahwa data tersebut dapat digunakan sebagai kunci utama. Pada contoh tabel keahlian dibuatkan kunci alternatif diberi nama kode_keahlian. Kunci alternatif yaitu dibuat seolah-olah ada untuk menandai setiap data dari Nama_Keahlian.

Tabel 7.2: Tabel Keahlian

Nama_Keahlian	Deskripsi
Data Analyst	Menganalisa kebutuhan pengguna
Coding	Membuat kode program
Tester	Melakukan testing program

Dimisalkan Data Analyst dibuatkan kode_keahlian DA, atau Coding dibuatkan kode_keahlian C dan Tester dibuatkan kode_keahlian T. Kode_keahlian ini dibuat seolah-olah ada dibuatkan sebagai kunci utama, maka kode keahlian ini disebut sebagai kunci alternatif.

5. Komposit key

Komposit key adalah menggabungkan lebih dari satu atribut untuk mendapatkan data yang unik.

6. Foreign key

Foreign key adalah sebuah atribut yang digunakan sebagai kunci tamu pada sebuah relasi di mana atribut tersebut merupakan kunci utama pada atribut yang lain.

7. Kardinalitas pemetaan

Kardinalitas pemetaan adalah banyaknya elemen pada tiap entitas yang dapat berelasi dengan elemen entitas lain dalam satu relasi. Adapun jenis-jenis kardinalitas meliputi :

- Hubungan satu ke satu atau *one to one* atau 1:1, yaitu menghubungkan maksimal satu elemen dari entitas A ke satu elemen ke entitas B. Begitu juga sebaliknya maksimum satu elemen dari entitas B ke maksimum satu ke entitas A. Sebagai contoh adalah hanya seorang rektor yang bertugas pada sebuah universitas, begitu juga sebaliknya jika dibaca yaitu sebuah universitas dikepalai oleh satu orang rektor.



Gambar 7.4: Relasi *one to one*

- Hubungan satu ke banyak atau *one to many* atau 1:N, yaitu menghubungkan maksimal satu elemen dari entitas A ke maksimal banyak elemen ke entitas B. Sebagai contoh adalah entitas bagian dan pegawai. Satu bagian pasti mempekerjakan minimal satu pegawai dan maksimal banyak pegawai. Satu pegawai pasti dipekerjakan pada satu bagian.



Gambar 7.5: Relasi *one to many*

- c. Hubungan banyak ke satu atau *many to one* atau *N:1*, yaitu menghubungkan maksimal banyak elemen dari entitas A ke maksimal satu elemen ke entitas B. Untuk contoh *may to one* dapat dilihat dari contoh *one to many*, jadi posisi entitas bagian dan pegawai dibalik. Jadi pembacaannya menjadi, satu pegawai pasti bekerja pada satu bagian. Dan satu bagian mempekerjakan lebih dari satu pegawai.



Gambar 7.6: Relasi *many to one*

- d. Hubungan banyak ke banyak atau *many to many* atau *N:M*, yaitu menghubungkan maksimal banyak elemen dari entitas A ke maksimal banyak elemen ke entitas B. Sebagai contoh adalah mahasiswa dan matakuliah. Satu mahasiswa dapat mengikuti satu atau lebih dari satu matakuliah. Dan matakuliah dapat diikuti oleh satu atau lebih dari satu mahasiswa.



Gambar 7.7: Relasi *many to many*

7.3 Tahapan pembuatan ERD

Untuk mendapatkan Entity Relationship Diagram atau ERD yang baik maka dibuat secara bertahap. Terdapat dua kelompok tahapan penting dalam pembuatan ERD, yaitu :

1. Tahapan pembuatan ERD awal atau di sebut dengan *preliminary desain*. Pada tahap ini dibuat ERD secara sederhana dan minimal.
2. Tahapan berikutnya adalah tahap optimasi pembuatan ERD, yaitu melakukan koreksi terhadap tahapan awal. Pada tahap ini dilakukan proses dekomposisi, kardinalitas dan penambahan relasi. (Suryana, 2012)

Dari dua kelompok tahapan tersebut dapat dijabarkan tahapan awal pembentukan ERD hingga optimasi pembentukan ERD. Untuk lebih mudah pemahaman tahapan pembuatan ERD, maka akan diilustrasikan pada sebuah perancangan basis data dalam pembuatan sebuah aplikasi sederhana. Berikut ini adalah tahapan-tahapan dalam pembuatan ERD (Simarmata, 2007):

1. Menentukan Entitas
2. Menentukan Relasi
3. Menggambarkan ERD sederhana untuk sementara
4. Mengisi kardinalitas
5. Tentukan kunci utama
6. Gambar ERD berdasarkan kunci
7. Menentukan atribut
8. Pemetaan atribut
9. Gambar ERD dengan atribut
10. Periksa hasil

7.4 Contoh Kasus

Sebagai contoh adalah perancangan basis data dalam aplikasi rental atau penyewaan film. Untuk dapat menyewa film, setiap orang harus terdaftar sebagai member. Setiap pelanggan yang ingin mendaftar sebagai member

hanya perlu memberikan data identitas diri. Setiap member dapat menyewa film lebih dari 1 dan maksimal 5 film dalam satu nota atau tidak menyewa film sama sekali. Setiap film yang dipinjam diberikan wakat sewa 2 hari. Setiap film wajib memiliki satu kategori sebagai penanda jenis film.

Sesuai dengan contoh kasus tersebut maka akan dibuat ERD berdasarkan tahapan yang sudah dijelaskan sebelumnya.

1. Menentukan Entitas

Pada kasus ini yang menjadi entitas adalah Member, Film dan Kategori.

2. Menentukan Relasi

Langkah awalnya adalah membuat matrik relasi entitasnya :

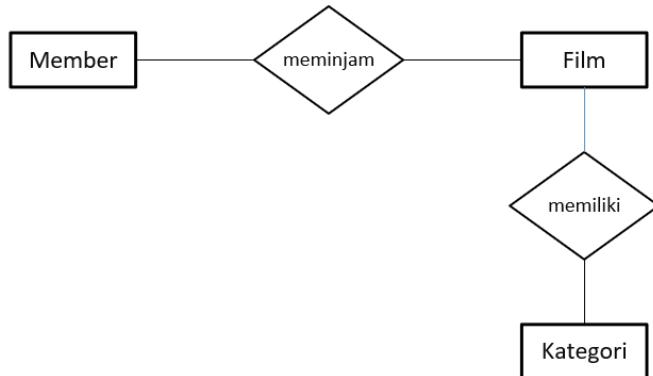
Tabel 7.3: Tabel penentuan relasi

	Member	Film	Kategori
Member		dipinjam	
Film	meminjam		dimiliki
Kategori		memiliki	

Pada tabel 7.8 dapat dijelaskan bahwa film dipinjam oleh member, di mana dipinjam adalah nama relasi yang menghubungkan entitas film dengan member. Member meminjam film, di mana meminjam adalah nama relasi yang menghubungkan entitas member dengan film.

3. Menggambarkan ERD sederhana

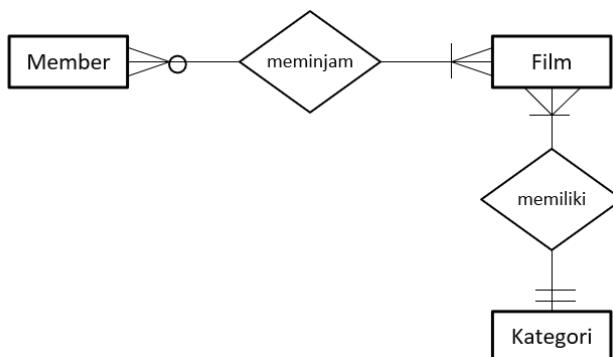
Proses ini dikerjakan berdasarkan matriks pada tabel penentuan relasi.

**Gambar 7.9:** ERD sederhana

4. Mengisi kardinalitas

Berdasarkan penjabaran kasus diatas maka dapat disimpulkan sebagai berikut :

- Setiap satu member dapat meminjam satu film atau maksimal 5 film
- Tidak semua film dipinjam oleh member, tetapi satu film dapat dipinjam oleh banyak member
- Setiap film pasti memiliki satu kategori
- Setiap satu kategori pasti dimiliki oleh minimal satu film dan maksimal banyak film

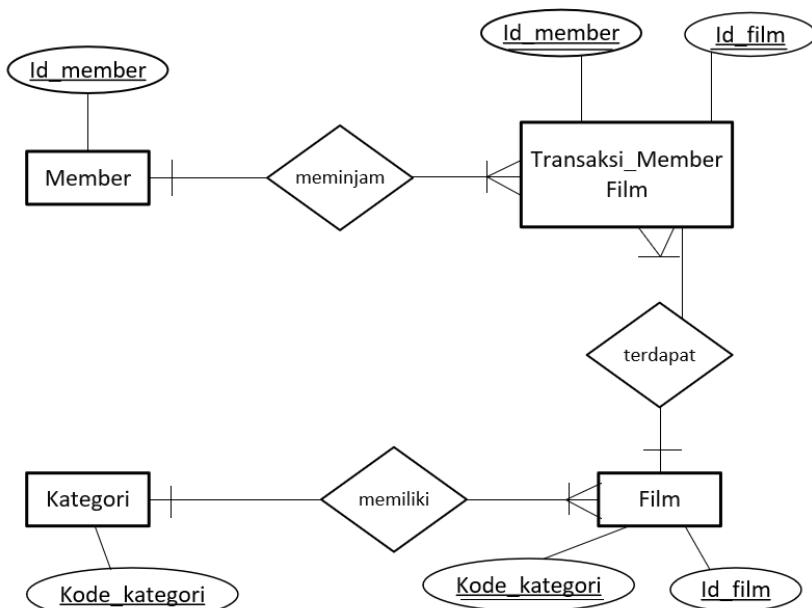
**Gambar 7.10:** ERD dengan kardinalitas

5. Tentukan kunci utama

Terdapat tiga entitas yaitu entitas Member, Film dan Kategori. Pada entitas Member yang menjadi kunci utama adalah id_member, entitas Film yang menjadi kunci utama adalah id_film dan entitas Kategori yang menjadi kunci utama adalah kode_kategori.

6. Gambar ERD berdasarkan kunci

Terdapat relasi *many-to-many* pada gambar ERD sebelumnya, yaitu entitas member dengan entitas film. Karena relasi *many-to-many* maka akan dibuat entitas baru diantara entitas member dan film. Kunci utama untuk member-film adalah gabungan antara id_member dan id_film.



Gambar 7.11: ERD dengan berdasarkan kunci

7. Menentukan atribut

Untuk melengkapi entitas maka diperlukan atribut lain untuk menampung properti dari masing-masing entitas.

Entitas Member : id_member, nama_member, alamat. Dan sesuai dengan kebutuhan pengguna basis data, yang menjadi data id_member adalah identitas diri seperti nomor KTP.

Entitas Film : id_film, nama_film

Entitas Kategori : kode_kategori, nama_kategori

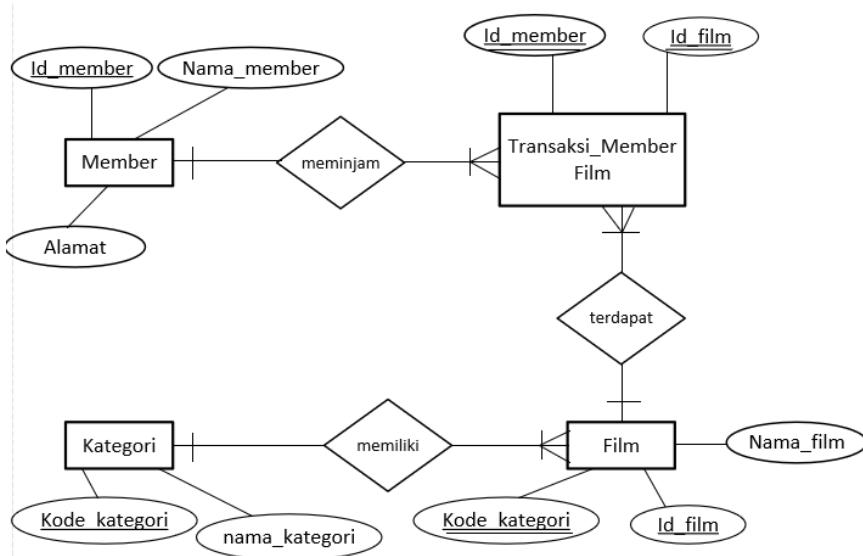
Penentuan atribut disesuaikan dengan kebutuhan pengguna, misalnya untuk entitas film, pengguna membutuhkan data sinopsis singkat film berarti perlu ditambahkan satu atribut lagi untuk menampung data tersebut.

8. Pemetaan atribut

Atribut	Entitas
Nama_member	Member
Alamat	Member
Nama_film	Film
Nama_kategori	Kategori

9. Gambar ERD dengan atribut

Setelah selesai pemetaan atribut, maka atribut dipasangkan pada ERD sesuai dengan pemetaannya. Simbol *primary key* dan *foreign key* harus disesuaikan.



Gambar 7.12: ERD dengan atribut

10. Periksa hasil

Setelah semua terselesaikan, maka dilakukan pemeriksaan terakhir kesesuaian kebutuhan dengan hasil akhir.

Bab 8

Arsitektur Pemodelan Data

8.1 Pemodelan data dan Model Data

Pemodelan data (*data modeling/ data modelling*) merupakan langkah awal dalam proses desain *database*, yang juga merupakan tahap paling tinggi dan abstrak, juga sering disebut dengan *Conceptual Data Model* (Watt, 2014). Tujuan dari fase ini adalah untuk dapat menjelaskan data-data yang terdapat dalam *database* (misalnya entitas : mahasiswa, guru, dosen, mata kuliah, mata pembelajaran), hubungan antar item (misalnya: mahasiswa diawasi dosen, dosen mengajar mata kuliah) serta batasan pada data (misalnya: NIM mahasiswa bernilai tepat 8 digit, mata kuliah memiliki nilai SKS sebanyak 2,3,4 dan 5). Pemodelan data dapat membantu untuk memahami dalam representasi visual data serta mematuhi aturan-aturan bisnis, kepatuhan terhadap peraturan, dan kebijakan yang diterapkan terhadap data terkait.

Model data adalah spesifikasi struktur data dan aturan bisnis yang mewakili kebutuhan bisnis(Sherman, 2015). Salah satu hal pertama yang dapat dilakukan saat mengembangkan aplikasi adalah berbicara dengan grup bisnis tentang kebutuhan sistem yang mereka inginkan, dan kemudian mendokumentasikannya. Selain kebutuhan sistem tersebut, akan ada sekumpulan data yang dibutuhkan untuk menjalankan aplikasi tersebut, baik

berupa sistem pelaporan, analisis, maupun pemrosesan transaksi. Selain data yang diperlukan ini, juga akan terdapat beberapa aturan bisnis yang menentukan hubungan antara berbagai entitas data atau objek yang diperlukan.

Pandangan tradisional terhadap pemodelan data dimulai dari pemodelan data logis (*Logical Data Modeling*) (James V Luisi, 2014). *Logical Data Modeling* adalah bagian dari siklus hidup pengembangan sistem / *Software Development Life Cycle* (SDLC) untuk pengembangan aplikasi dan bertujuan untuk dapat meminimalisir redundansi data dengan proses perancangan *database* yang dikenal dengan istilah normalisasi. Tentunya terdapat perbedaan antara *Logical Data Model* dengan *Logical Data Architecture* (LDA) yang merupakan kondisi khusus dalam bisnis untuk mengatur seluruh data perusahaan, terlepas dari apakah LDA tersebut akan berpartisipasi dalam *database* untuk otomatisasi ataupun tidak.

Manfaat yang diharapkan dari penggunaan pemodelan data dalam hal ini untuk mencapai redundansi minimal adalah pemodelan data diharapkan untuk dapat menyederhanakan logika aplikasi dengan hanya mempertahankan nilai bisnis dalam sebuah kolom *database* tertentu dalam sebuah tabel *database*, dibandingkan dengan harus menulis kode aplikasi untuk mempertahankan nilainya di beberapa tabel *database* berbeda. Model data sendiri didefinisikan sebagai model abstrak yang mengatur deskripsi data, semantik data dan batasan dari konsistensi data. Model data menekankan terhadap segala data yang dibutuhkan dan bagaimana data itu harus diatur daripada operasi apa saja yang akan dilakukan pada data tersebut. Model data diibaratkan seperti rencana bangunan seorang arsitek, yang membantu membangun *Conceptual Data Model* dan mengatur hubungan antar item data.

Model data memberikan kerangka kerja untuk data yang akan digunakan dalam sistem informasi dengan memberikan definisi dan format khusus. Jika model data digunakan secara konsisten pada sistem secara keseluruhan, maka aplikasi yang berbeda dapat berbagi data dengan baik. Hasil dari proses tersebut dapat digambarkan kedalam sebuah diagram. Dilain sisi, pembangunan sistem serta pembangunan *interface* seringkali membutuhkan biaya yang besar serta kesulitan dalam pengoperasian dan pemeliharaan. Terkadang hal ini membatasi bisnis daripada mendukungnya. Hal ini terjadi ketika kualitas model data yang diterapkan dalam sistem dalam kondisi buruk (West, 2011). Dua tipe dari teknik model data adalah Entity Relationship (E-R) Model dan Unified Modelling Language (UML).

8.2 Tujuan Utama dari Pemodelan Data

Model data yang dirancang dengan baik adalah landasan untuk membangun aplikasi *business intelligence* (BI), *data warehousing* (DW) ataupun aplikasi bisnis sehingga dapat memberikan nilai bisnis yang signifikan (Sherman, 2015). Hasil pemodelan data yang efektif dalam mengubah data menjadi aset informasi perusahaan yang konsisten, komprehensif, dan terkini. Data diubah dari sistem operasional atau sumber menjadi *data warehouse*, *data mart*, atau *online analytical processing* (OLAP) untuk dapat dilakukan analisis.

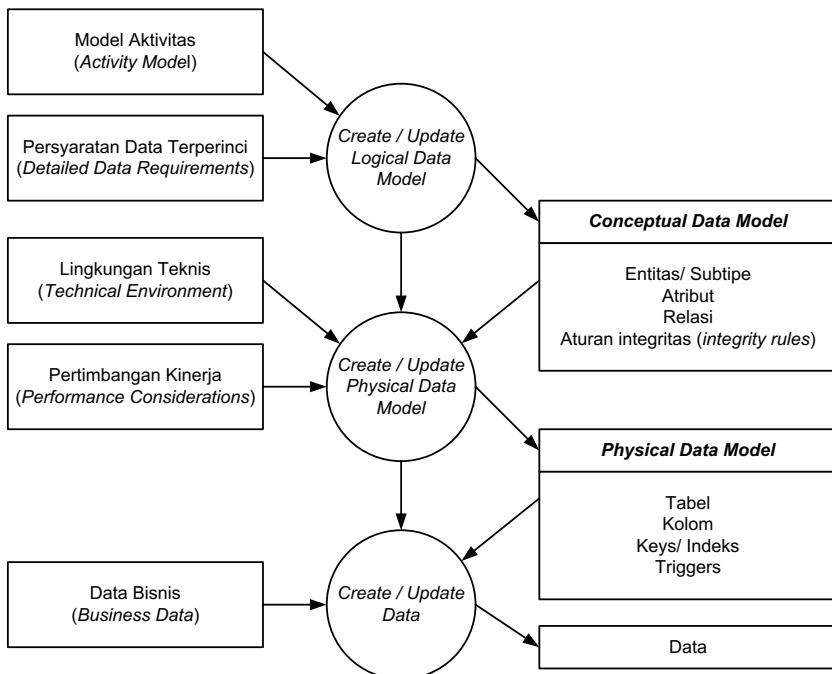
Sehingga, tujuan utama dari penggunaan model data adalah sebagai berikut:

1. Seluruh objek data yang dibutuhkan oleh *database* dipastikan dapat terwakili secara akurat. Penghilangan data akan menyebabkan pembuatan laporan yang salah dan menghasilkan hasil yang salah.
2. Model data dapat membantu merancang *database* pada tingkat konseptual (*Conceptual Data Model / CDM*), logis (*Logical Data Model/ LDM*), dan fisik (*Physical Data Model/ PDM*).
3. Struktur dari model data membantu untuk menentukan relasional antar tabel, *primary keys*, *foreign keys* dan prosedur yang tersimpan (*stored procedures*).
4. Model data menghadirkan gambaran yang jelas tentang data awal dan dapat digunakan oleh *database developers* untuk membuat *database* fisik.
5. Model data juga dapat membantu dalam mengidentifikasi data yang hilang dan berlebihan
6. Meskipun pembuatan awal model data membutuhkan waktu dan tenaga, namun pada jangka panjang, hal ini akan membuat *upgrade* dan *maintenance* dari infrastruktur IT lebih murah dan cepat.

8.3 Tingkatan Pemodelan Data

Terdapat tiga tingkatan model data, yang tumbuh semakin kompleks, yaitu konseptual/ *Conceptual Data Model* (CDM), logis/ *Logical Data Model* (LDM), dan fisik/ *Physical Data Model* (PDM). *Conceptual Data Model*

mewakili pandangan bisnis tingkat tinggi yang sangat efektif untuk berkomunikasi dengan grup bisnis untuk memahami dan menentukan proses bisnis di dalam perusahaan. *Logical Data Model* adalah tingkatan berikutnya yang membantu memahami detail data, tetapi tidak bagaimana penerapannya. *Logical Data Model* adalah pandangan arsitek atau desainer dari data. *Physical Data Model* membentuk tingkat ketiga. *Logical Data Model* digunakan sebagai *blueprint* dari data apa yang terlibat sedangkan *Physical Data Model* merinci bagaimana data tersebut akan diimplementasikan. Kemudian *database administrator* dan *developer* sistem akan mengubah *Logical Data Model* ke dalam tabel, kolom, kunci, dan entitas fisik database lainnya.



Gambar 8.1: Data Modeling (West, 2011)

8.3.1 Conceptual Data Model

Model Data Konseptual/ *Conceptual Data Model* (CDM) adalah konsep terkait dengan pandangan pemakaian terhadap data yang disimpan dalam *database*(Setyarini, 2016)(A.S., Rosa dan Shalahudin, 2013). *Conceptual Data Model* dibuat kedalam bentuk tabel-tabel tanpa memberikan keterangan tipe data, namun tetap menggambarkan relasi antar tabel untuk keperluan implementasi ke dalam *database*. *Conceptual Data Model* menggunakan sistem simbol standar yang membentuk bahasa formal meskipun tidak rumit, yang mengomunikasikan banyak pengetahuan tentang informasi yang dimodelkan(James V Luisi, 2014).

Bahasa visual ini efektif untuk menyampaikan pandangan pengguna bisnis tentang data yang sedang mereka kerjakan. *Conceptual Data Model* berfokus pada identifikasi data yang digunakan dalam bisnis tetapi tidak pada aliran pemrosesan atau karakteristik fisiknya. Perspektif *Conceptual Data Model* tidak bergantung pada aplikasi bisnis yang mendasarinya(Sherman, 2015). Sistem simbol yang digunakan dalam *Conceptual Data Model* meminjam sejumlah konstruksi pemodelan dasar yang ditemukan dalam *Entity Relationshhip Diagram* (ERD), yang berisi entitas, atribut, dan relasi.

Karakteristik dari *Conceptual Data Model* adalah sebagai berikut:

1. Tujuan dari *Conceptual Data Model* adalah untuk mengomunikasikan pengetahuan bisnis kepada setiap individu yang tidak familiar dengan bisnis tersebut.
2. Ruang lingkup *Conceptual Data Model* adalah dari perspektif area subjek bisnis data, sebagai lawan dari lingkup proyek otomasi, aplikasi otomasi, *database* otomasi, atau interface otomatisasi.
3. *Conceptual Data Model* menawarkan cakupan seluruh organisasi dari konsep bisnis.
4. *Conceptual Data Model* dirancang dan dikembangkan untuk audiens bisnis.
5. *Conceptual Data Model* dikembangkan secara independen dari spesifikasi perangkat keras seperti kapasitas penyimpanan data, lokasi atau spesifikasi perangkat lunak seperti vendor dan teknologi DBMS. Fokusnya adalah untuk merepresentasikan data sebagaimana pengguna akan melihatnya di "dunia nyata".

6. Nama-nama objek dalam *Conceptual Data Model* dibatasi secara ketat pada bahasa yang digunakan dalam bisnis, tidak termasuk semua terminologi teknis yang terkait dengan istilah khusus pada otomasi.
7. Aturan pembuatan diagram adalah yang menekankan pada apa yang dapat dilihat dan dipahami seseorang dengan nyaman pada halaman pribadinya.
8. Titik data bisnis secara sederhana dikaitkan dengan objek data yang akan dimilikinya dan tidak diambil melalui proses rekayasa data yang disebut "normalisasi" untuk memisahkan atribut ke dalam tabel kode.
9. Abstraksi data, seperti merujuk ke objek bisnis dengan cara yang lebih umum dan umum, tidak dilakukan karena sering kali kehilangan maksud bisnis dan kemudian menjadi kurang dapat dikenali oleh bisnis.
10. Detail teknis, yang sering ditemukan dalam ERD, seperti opsionalitas dan kardinalitas numerik tertentu dapat dihilangkan.

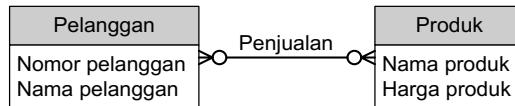
Pendekatan modern untuk *Conceptual Data Model* adalah dengan menggabungkannya sebagai perluasan dari LDA. Faktanya, setiap *Conceptual Data Model* harus sesuai dengan satu area subjek bisnis data dan harus dikembangkan oleh pengguna bisnis yang telah dibimbing oleh ahli informasi bisnis untuk membantu dalam pemeliharaan LDA.

Tiga hal mendasar yang terdapat dalam *Conceptual Data Model* adalah sebagai berikut:

1. Entitas: Hal dunia nyata
2. Atribut: Karakteristik atau properti dari suatu entitas
3. Hubungan/ Relasi: Ketergantungan atau asosiasi antara dua entitas

Contoh *Conceptual Data Model* sederhana yang mengambil contoh pada *database penjualan*, dapat dimisalkan sebagai berikut:

1. Pelanggan dan Produk adalah entitas.
2. Nomor pelanggan dan nama pelanggan adalah atribut dari entitas Pelanggan.
3. Nama produk dan harga adalah atribut dari entitas produk.
4. Penjualan adalah hubungan/ relasi antara pelanggan dan produk.



Gambar 8.2: Contoh Sederhana dari *Conceptual Data Model* pada *Database Penjualan*

Contoh *Conceptual Data Model* lainnya, yang mengambil contoh pada *database upload video*, dapat dimisalkan sebagai berikut:

1. Video, Komentar, Label, *Upload History*, Lokasi dan Album adalah entitas
2. ID video, judul video, deskripsi video, privasi video, nama pengunggah video, no hp pengunggah video, surel pengunggah video, alamat pengunggah video dan view video adalah atribut dari entitas Video.
3. Tanggal komentar dan konten komentar adalah atribut dari entitas Komentar.
4. Judul label adalah atribut dari entitas Label.
5. Tanggal upload adalah atribut dari entitas Upload History.
6. Nama lokasi dan alamat lokasi adalah atribut dari entitas Lokasi.
7. Judul album, deskripsi album dan view album adalah atribut dari entitas Album



Gambar 8.3: Contoh dari *Conceptual Data Model* pada *Database Upload Video*

8.3.2 Logical Data Model

Model data logis / *Logical Data Models* (LDM) lebih bersifat teknis dan sering mewakili ruang lingkup data untuk upaya atau proyek otomatisasi tertentu(James V Luisi, 2014). *Logical Data Models* adalah yang paling banyak digunakan dalam merancang aplikasi Business Intelligence(Sherman, 2015). *Logical Data Models* dibangun berdasarkan persyaratan yang diberikan oleh grup bisnis. *Logical Data Models* mencakup tingkat detail yang lebih lanjut, mendukung persyaratan lain yang terkait dengan sistem bisnis dan data *Logical Data Models* termasuk dalam fase desain logis (*logical design phase*) sebagai langkah rekayasa data dalam SDLC.

Logical data models juga memanfaatkan sistem simbol standar yang membentuk bahasa formal yang tidak terlalu rumit, yang mengomunikasikan pengetahuan. Meskipun demikian, tidak seperti diagram *conceptual data models* yang mudah dibaca, *logical data models* cenderung lebih sulit dipahami oleh orang awam. *Logical data models* efektif namun untuk mengomunikasikan pandangan perancang informasi kepada analis bisnis pada tim pengembangan aplikasi dan administrator *database* yang akan melakukan fase desain *database* fisik. Setelah analis bisnis mengkonfirmasi bahwa *Logical data models* memenuhi semua persyaratan data, administrator *database* kemudian bebas untuk melakukan fase desain fisik.

Karakteristik dari *Logical Data Model* adalah sebagai berikut:

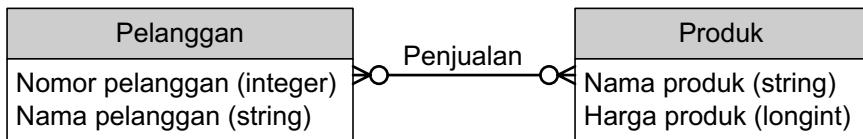
1. Tujuan *Logical Data Model* adalah untuk mengomunikasikan kepada software developers sebagai suatu bentuk detail dari persyaratan data untuk mendorong desain *database*
2. Lingkup dari *Logical Data Model* biasanya dari perspektif proyek otomasi, aplikasi otomatisasi, *database* otomasi, atau interface otomatisasi.
3. Dirancang dan dikembangkan secara independen dari DBMS.
4. Atribut data akan memiliki tipe data dengan presisi dan panjang yang tepat.
5. Proses normalisasi ke model diterapkan biasanya hingga 3NF.
6. Nama objek dalam model pada *Logical Data Model* termasuk terminologi teknis yang terkait dengan istilah khusus dalam

otomatisasi, seperti penggunaan kata-kata (misalnya, jenis, file batch, antarmuka, dan catatan kontrol sistem).

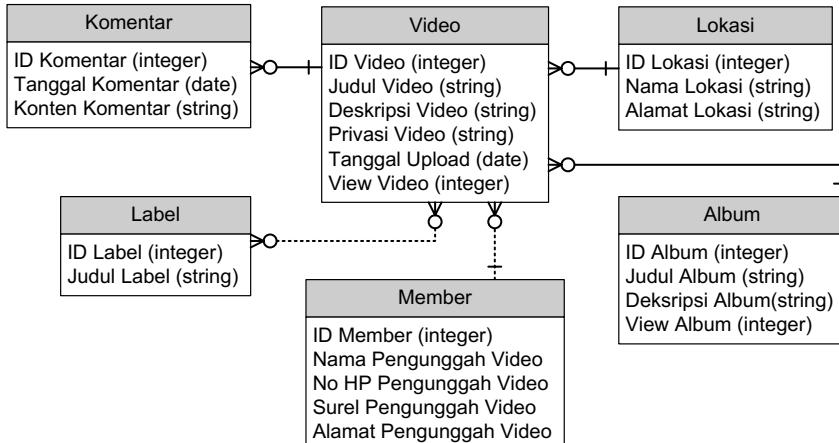
7. Konvensi pembuatan diagram pada *Logical Data Model* sering kali memerlukan pakar teknis yang telah dilatih untuk bekerja dengan struktur "*bill-of-material*" dan "*subtypes*".
8. Titik data bisnis pada *Logical Data Model* diambil melalui proses rekayasa data yang disebut "normalisasi".
9. Abstraksi data pada *Logical Data Model* seperti merujuk ke objek bisnis yang lebih umum dan dengan cara yang umum terkait praktik yang sering dilakukan.
10. Detail teknis pada *Logical Data Model* sering ditemukan dalam ERD, seperti opsionalitas dan kardinalitas numerik tertentu yang diperlukan.

Logical Data Model menambahkan informasi lebih lanjut ke elemen *Conceptual Data Model*. Keuntungan menggunakan *Logical Data Model* adalah memberikan landasan untuk membentuk dasar bagi *Physical Data Model*, namun, struktur pemodelannya tetap umum. Pada *Logical Data Model* tidak mendefinisikan primary key ataupun secondary key.

Merujuk pada contoh terkait *Conceptual Data Model* yang telah dipaparkan sebelumnya, untuk pembangunan pada *Logical Data Model* dapat dilengkapi dengan tipe data, sehingga dapat bertransformasi sebagai berikut:



Gambar 8.4: Contoh Sederhana dari *Logical Data Model* pada *Database Penjualan*



Gambar 8.5: Contoh dari *Logical Data Model* pada *Database Upload Video*

8.3.3 Physical Data Model

Model Data Fisik/ *Physical Data Model* (PDM) adalah model yang menggunakan beberapa tabel untuk menggambarkan data dan hubungan antar kolom(Setyarini, 2016)(A.S., Rosa dan Shalahudin, 2013). Setiap tabel mempunyai sejumlah kolom yang memiliki nama yang unik beserta tipe datanya masing-masing. Menerapkan *Physical Data Model* membutuhkan pemahaman tentang karakteristik dan batasan kinerja dari DBMS yang digunakan(Sherman, 2015). Dalam mengembangkan *Physical Data Model* harus memahami bagaimana tabel, kolom, tipe data, dan hubungan antara tabel dan kolom diimplementasikan dalam DBMS tertentu.

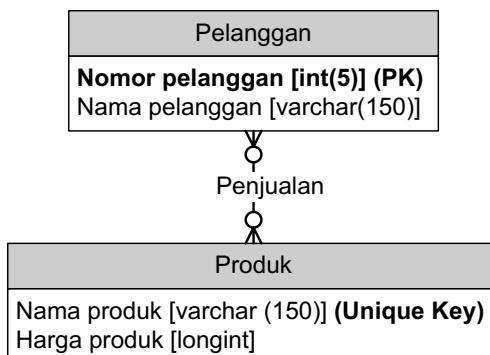
Physical data model adalah yang paling teknis dari semua model data, dan dari berbagai kriteria masuk yang dikaitkan dengannya sebelum dapat dimulai, yang paling penting adalah analisis jalur transaksi/ *Transaction Path Analysis* (TAPA)(James V Luisi, 2014). Desain *database* fisik adalah sekumpulan tugas yang dilakukan oleh *database administrator* (DBA) yang memerlukan pengetahuan mendalam tentang produk DBMS tertentu dan lingkungan fisik *database* untuk menentukan struktur *database* fisik yang sesuai dan parameter yang paling dapat memenuhi persyaratan kinerja *database* untuk satu atau lebih aplikasi. Pengetahuan DBA tentang DBMS dapat dianalogikan dengan pengetahuan dokter tentang kedokteran, dan oleh karena itu pasien yang harus

dianalisis DBA adalah pemrograman aplikasi, sedangkan TAPA dapat dianalogikan juga dengan pasien. Informasi yang dibutuhkan DBA adalah informasi tentang aplikasi sehingga akan sangat merugikan jika desain *database fisik* tidak sesuai dengan yang dibutuhkan.

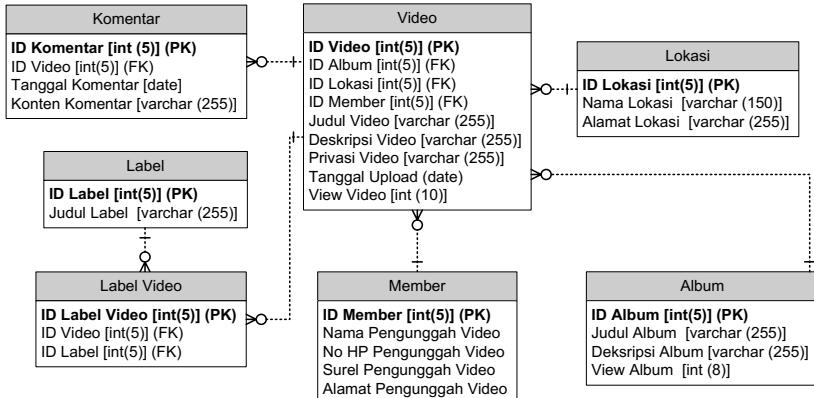
Karakteristik model data fisik:

1. *Physical data model* menggambarkan kebutuhan data untuk satu proyek atau aplikasi meskipun mungkin terintegrasi dengan model data fisik lainnya berdasarkan ruang lingkup proyek.
2. *Physical data model* berisi hubungan antara tabel yang membahas kardinalitas dan nullabilitas hubungan.
3. Dikembangkan untuk versi tertentu dari DBMS, lokasi, penyimpanan data atau teknologi yang akan digunakan dalam proyek.
4. Kolom harus memiliki tipe data yang tepat, panjang yang ditetapkan dan nilai default.
5. *Primary key* dan *foreign key*, tampilan, indeks, profil akses, otorisasi dan lain-lain, dapat ditentukan.

Merujuk pada contoh terkait *Logical Data Model* yang telah dipaparkan sebelumnya, untuk pembangunan pada *Physical Data Model* bertujuan untuk dapat memvisualisasikan struktur *database* dengan mereplikasi kunci kolom *database*, batasan, indeks, pemicu, dan fitur RDBMS, sehingga dapat bertransformasi sebagai berikut:



Gambar 8.6: Contoh Sederhana dari *Physical Data Model* pada *Database Penjualan*



Gambar 8.7: Contoh dari *Physical Data Model* pada *Database Upload Video*

8.4 Alur Kerja Pemodelan Data

Alur kerja pemodelan data dimulai dari tahap persyaratan bisnis hingga implementasi fisik database. Dari fase awal hingga fase akhir, proses pemodelan data adalah sebagai berikut:

1. Kumpulkan kebutuhan bisnis (*business requirements*).
2. Analisis data yang dibutuhkan oleh *business requirements*.
3. Mengidentifikasi hubungan data.
4. Buat berbagai model data yang dibutuhkan.
5. *Conceptual Data Model*
6. *Logical Data Model*
7. *Physical Data Model*
8. Mendukung pengembangan aplikasi.
9. Buat spesifikasi aplikasi
10. Kembangkan aplikasi
11. Menerapkan aplikasi

Alur kerja pemodelan data dimulai dengan pengumpulan syarat-syarat bisnis (*business requirements*). Akan terjadi negosiasi antara *developer* sistem dengan grup bisnis dan menanyakan apa yang dibutuhkan untuk aplikasi tersebut. Setelah mengumpulkan *business requirements* tersebut, *developer* sistem dapat mengembangkan *Conceptual Data Model*. Untuk tahap ini, *developer* sistem perlu mengetahui subjek bisnis apa yang ditangkap dalam persyaratan data, apa artinya, di mana datanya, dan apa saja hubungan dalam data tersebut.

Kemudian *developer* sistem membuat *Logical Data Model*, yang merupakan sebagian besar pekerjaan pemodelan data. Ini mencakup mengambil apa yang *developer* sistem pelajari dalam tahap pengumpulan kebutuhan bisnis dan *Conceptual Data Model* serta memahami detail entitas dan hubungan yang ada dalam *Conceptual Data Model*. Dari sini, *developer* sistem membuat spesifikasi aplikasi. Inti dari pembuatan *Logical Data Model* adalah membuat *blueprint* yang memandu pengembangan kode aplikasi dan memastikannya berfungsi dengan benar. Setelah membuat *Logical Data Model* dan spesifikasi aplikasi, langkah selanjutnya adalah merancang *Physical Data Model*. Kegiatan ini termasuk menambahkan partisi, pengindeksan, *tablespaces*, dan apa pun yang diperlukan untuk menghasilkan kinerja yang optimal. Ini diikuti dengan mengembangkan aplikasi secara fisik.

Mengembangkan *Logical Data Model* adalah proses yang iteratif. Setelah *developer* sistem mulai pengembangan aplikasi, *developer* sistem mungkin menemukan hubungan, algoritma, atau dependensi baru yang tidak terlihat selama diskusi pertama *Logical Data Model*. Ini sangat umum dan bukan masalah besar. Proses iteratif ini merupakan penyempurnaan bertahap dari *Logical Data Model* dan *Physical Data Model* yang terjadi selama pengembangan dan pengujian aplikasi. Hasilnya adalah *Physical Data Model* tervalidasi yang digunakan untuk mengimplementasikan kebutuhan bisnis dan data dalam aplikasi.

8.5 Perbandingan Conceptual, Logical dan Physical Data Model terhadap E-R Model

Terdapat beberapa perbedaan antara masing-masing fase pada *Conceptual Data Model*, *Logical Data Model* dan *Physical Data Model*, dimana setiap fase tersebut memang berisikan entitas dengan atribut dan relasinya masing-masing, namun masih terdapat beberapa perbedaan yang dapat dilihat pada tabel sebagai berikut:

Tabel 8.1: Perbandingan *Conceptual Data Model*, *Logical Data Model* dan *Physical Data Model* terhadap E-R Model

Fitur E-R	<i>Conceptual Data Model</i>	<i>Logical Data Model</i>	<i>Physical Data Model</i>
Nama Entitas	Ya	Ya	Ya
Relasi	Ya	Ya	Ya
Kolom	Tidak	Ya	Ya
Tipe Kolom	Tidak	Opsional	Ya
Primary Key	Tidak	Tidak	Ya
Foreign Key	Tidak	Tidak	Ya

8.6 Keunggulan dan Kelemahan dari Model Data

Model data telah terbukti memiliki banyak manfaat dalam pengembangan sebuah sistem. Keunggulan dari penggunaan model data adalah sebagai berikut:

1. Tujuan utama perancangan model data adalah untuk memastikan bahwa objek data yang ditawarkan oleh tim fungsional terwakili secara akurat.
2. Model data harus cukup rinci untuk digunakan dalam membangun *database fisik*.
3. Informasi dalam model data dapat digunakan untuk menentukan hubungan antara tabel, *primary key* dan *foreign key*, dan prosedur yang tersimpan.
4. Model data membantu bisnis untuk mengomunikasikan di dalam dan di seluruh organisasi.
5. Model data membantu mendokumentasikan pemetaan data dalam proses ETL (*Extract, Transform, Load*)
6. Membantu untuk mengenali sumber data yang benar untuk mengisi model.

Selain memiliki sisi keunggulan dari penggunaan model data, berikut adalah kekurangan dalam penggunaan model data, sebagai berikut:

1. Untuk mengembangkan model data seseorang harus mengetahui karakteristik penyimpanan data fisik.
2. Model data adalah sistem navigasi yang menghasilkan pengembangan aplikasi yang kompleks, sehingga dibutuhkan pengetahuan tentang kebenaran biografis.
3. Perubahan kecil pada struktur menimbulkan modifikasi di seluruh aplikasi.

Bab 9

Perancangan Basis Data Perpustakaan

9.1 Pengenalan Basis Data Perpustakaan

Sebelum kita mengetahui apa itu basis data perpustakaan dan bagaimana proses perancangannya. Sebaiknya mengetahui peran dan fungsi dari perpustakaan sehingga mengapa perlu dibuatkan basis data perpustakaan. Perpustakaan merupakan salah satu sarana yang sangat berperan penting dalam penyelenggaraan pendidikan disemua jenjang, baik itu mulai dari TK sampai dengan Perguruan Tinggi. Perpustakaan berfungsi bagi Pustaka sebagai sumber referensi dan bahan pustaka melalui koleksi buku-buku yang berguna dan sesuai dengan kurikulum pada saat itu (Salmubi, 2020).

Dalam memenuhi kebutuhan dalam memperoleh sumber referensi dan bahan Pustaka perlu dibuatkan pengelolaan data yang memadai dengan menggunakan sumber daya komputer. Sehingga mempermudah dalam melakukan proses administrasi baik itu proses penyimpanan atau create, pencarian atau read, perubahan atau update, dan penghapusan atau delete data perpustakaan yang dikenal dengan istilah CRUD dalam dunia sistem database.

Pada basis data perpustakaan ini merupakan suatu basis data yang bisa digunakan untuk mengelola data perpustakaan seperti data buku, data anggota, data petugas, data peminjaman, dan data pengembalian. Sehingga data tersebut dapat membantu dalam membangun sebuah sistem informasi perpustakaan apakah berbasis desktop maupun berbasis web. Nah adapun langkah-langkah membuat basis data perpustakaan tersebut diperlukan tahapan dalam merancangnya yang nanti akan dibahas di sub bab berikutnya.

9.2 Tujuan dan Manfaat Perancangan Basis Data Perpustakaan.

Alasan mengapa basis data perpustakaan ini perlu dipelajari kemudian bagaimana merancang basis data perpustakaan, tentu saja penulis membuat ini memiliki tujuan dan manfaat. Adapun tujuan membuat rancangan basis data perpustakaan sekolah ini antara lain :

1. sistem yang berjalan secara manual dapat direpresentasikan melalui aplikasi perangkat lunak sehingga mempermudah dalam melakukan pencatatan, pencarian, penyimpanan secara sistematis
2. Khususnya pada data perpustakaan, proses penyimpanan data perpustakaan disimpan disuatu database, sehingga data – data tersebut mudah dikelola dan menghindari dari kehilangan data.

Sedangkan manfaat dari perlunya mempelajari perancangan ini adalah sebagai berikut:

1. Bagi lingkup perpustakaan, kemudahan akses informasi data baik itu data buku, anggota, petugas dan juga mempercepat proses transaksi seperti proses peminjaman dan pengembalian, tentunya dengan adanya basis data akan sangat membantu. Hanya dengan mengetikan kode buku maka riwayat data buku yang sedang dipinjam atau yang sudah dikembalikan akan bisa dilihat datanya.
2. memudahkan pihak *system analyze* dalam mendeskripsikan sistem basis data perpustakaan secara kompleks dan juga pihak *programmer* saat membangun sistem informasi perpustakaan.

9.3 Tahapan dalam perancangan basis data perpustakaan

Adapun perancangan basis data perpustakaan dengan mengambil studi kasus perpustakaan sekolah sederhana, penulis buat disini dibagi menjadi 5 tahapan. Sebelum ke proses 5 tahapan tersebut diperlukan untuk mengetahui daftar penting dari rancangan basis data perpustakaan sekolah, antara lain :

1. Setiap perpustakaan sekolah pada umumnya akan berjalan jika memiliki data petugas buku, anggota, peminjaman, pengembalian dan rakbuku.
2. Buku dialokasikan di rak buku
3. Buku dapat dipinjam anggota perpustakaan lebih dari sekali
4. Peminjaman hanya diperbolehkan 2 buku saja
5. Pengembalian buku dilakukan sekali dari setiap peminjaman

Lima daftar penting yang telah diuraikan diatas berfungsi untuk dapat mendeskripsikan proses bisnis yang terjadi pada studi kasus perpustakaan sekolah. Oke, subbab berikut akan dilanjutkan dengan menjelaskan 5 tahapan dalam merancang basis data perpustakaan sekolah, antara lain :

9.3.1 Mengidentifikasi Entitas

Di bab sebelumnya telah dijelaskan tentang pengertian dari entitas yang merupakan suatu objek yang dapat diidentifikasi dalam lingkungan pemakai(Wati, 2013). Langkah awal dalam pembuatan *entity relationship diagram* yaitu mengidentifikasi entitas yang diperlukan dalam studi perpustakaan sekolah sederhana yaitu sebagai berikut :

1. Buku
2. Anggota
3. Rakbuku
4. Petugas

9.3.2 Mengidentifikasi Jenis Relasi

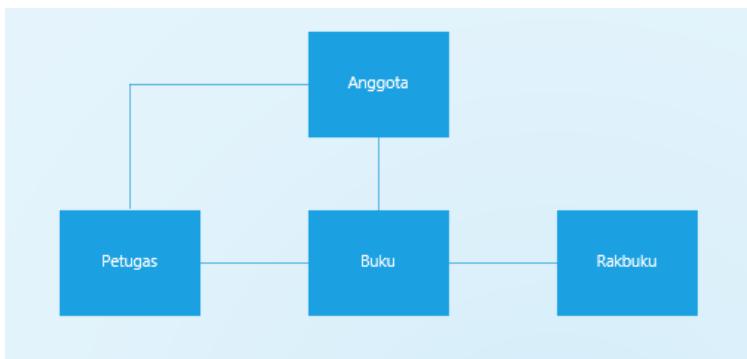
Jenis Relasi merupakan sekumpulan asosiasi antara satu atau beberapa jenis entitas yang berpartisipasi. Setiap jenis hubungan diberi nama yang menjelaskan fungsinya(Connolly and Begg, 2005). Untuk

mengidentifikasi jenis relasi diperlukan dalam tahapan kedua dalam pembuatan ERD, seperti yang terlihat pada tabel 1, yaitu :

Tabel 9.1: Tabel Mengidentifikasi Relasi Sistem Perpustakaan Sekolah

Entitas	Bentuk Relasi	Entitas
Anggota	Meminjam	Buku
Anggota	Mengembalikan	Buku
Petugas	Mengelola	Anggota
Petugas	Mengelola	Buku
Rakbuku	Memiliki	Buku

Diagram awal yang terbentuk jika dihubungkan secara entitas maka kurang lebihnya dapat dilihat pada gambar 1, yaitu :



Gambar 9.1: Diagram Awal terhubung secara Entitas

9.3.3 Menentukan Kardinalitas

Kardinalitas adalah relasi antar tabel memiliki rasio yang memiliki perbandingan jumlah baris pada satu tabel dengan tabel lainnya. Kardinalitas memiliki 3 jenis yaitu : one to one (1:1), one to many (1:M)

dan many to many (M:N) (Nur Hidayat, 2016). Berikut ini merupakan kardinalitas sesuai dengan studi kasus yang sedang dibahas, antara lain :

1. Anggota bisa meminjam buku, artinya kardinalitas anggota dengan buku berhubungan (M:N).



Gambar 9.2: Kardinalitas relasi meminjam buku

Dikarenakan pada entitas anggota dan buku terdapat relasi M:N pada meminjam maka akan terbentuk entitas komposit sehingga relasinya akan terbentuk :



Gambar 9.3: Kardinalitas relasi meminjam buku membentuk entitas komposit

2. Anggota mengembalikan buku , artinya kardinalitas anggota dengan buku berhubungan (M:N)



Gambar 9.4: Kardinalitas relasi mengembalikan buku

Begitu juga pada entitas anggota dan buku terdapat relasi M:N pada mengembalikan maka akan terbentuk entitas komposit sehingga relasinya akan terbentuk :



Gambar 9.5: Kardinalitas relasi mengembalikan buku membentuk entitas komposit

3. RakBuku memiliki buku, artinya kardinalitas rakbuku dengan buku berhubungan (1:1)



Gambar 9.6: Kardinalitas pada relasi memiliki rakbuku

4. Petugas mengelola buku lebih dari satu buku, artinya kardinalitas petugas dengan buku berhubungan (1:M)



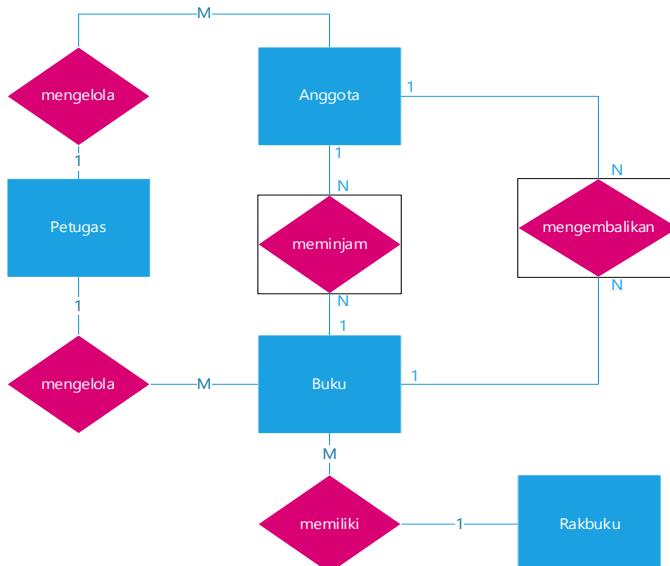
Gambar 9.7: Kardinalitas pada relasi mengelola buku

5. Petugas mengelola anggota lebih dari satu anggota, artinya kardinalitas petugas dengan anggota berhubungan (1:M)



Gambar 9.8: Kardinalitas pada relasi mengelola anggota

Dari hasil analisis tersebut gambaran ERD yang telah ditentukan kardinalitasnya seperti gambar 9.10 di bawah ini :



Gambar 9.10: ERD yang telah ditentukan kardinalitasnya

9.3.4 Mengidentifikasi Atribut dan Penentuan Primary Key

Langkah selanjutnya adalah identifikasi atributnya. Atribut yang dimaksud disini adalah property yang dimiliki oleh suatu entitas yang bermakna bagi organisasi(Nugroho, 2011). Atribut yang di identifikasi juga harus di lengkapi dengan primary key atribut tersebut untuk mewakili entitas yang di maksud. Tidak lupa juga sebagai catatan bahwa entitas yang mempunyai kardinalitas N tentunya memuat penghubung yang disebut sebagai foreign key. Adapun atribut dari entitas-entitas yang sudah di identifikasi sebelumnya sesuai pada studi kasus adalah dapat dilihat pada tabel 9.2 sebagai berikut :

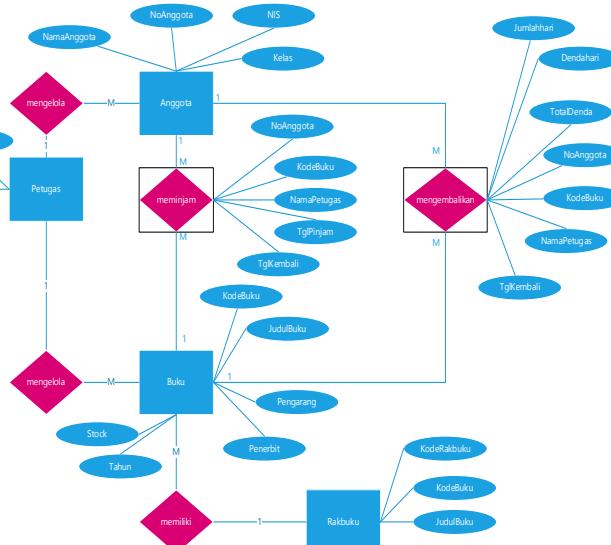
Tabel 9.2: Tabel identifikasi atribut dan penentuan primary key

Entitas	Attribute	Primary Key
Petugas	NamaPetugas	PK
	Password	
Anggota	NoAnggota	PK
	NamaAnggota	
	NIS	
	Kelas	
Buku	KodeBuku	PK
	JudulBuku	
	Pengarang	
	Penerbit	
	Tahun	
	Stock	
Rakbuku	NoRakbuku	PK
	KodeBuku	
	JudulBuku	
Peminjaman	NoAnggota	PK (Composite Key)
	KodeBuku	PK (Composite Key)
	NamaPetugas	PK (Composite Key)
	TanggalPinjam	

Pengembalian	NoAnggota	PK (Composite Key)
	KodeBuku	PK (Composite Key)
	NamaPetugas	PK (Composite Key)
	TanggalKembali	
	DendaHari	
	JumlahHari	
	TotalDenda	

9.3.5 Membuat ERD (Entity Relationship Diagram)

Entity-relationship diagramming (ERD) adalah teknik yang digunakan untuk memodelkan persyaratan data pada suatu organisasi, biasanya oleh analis sistem digunakan dalam persyaratan fase analisis proyek pengembangan sistem(Brady and Loonam, 2010). Pada tahapan kelima ini adalah tahapan terakhir yaitu dengan menyelesaikan ERD lengkapnya dengan atribut dan primary key yang bersifat penandaan atribut mana saja yang menjadi primary key dari tiap entitas. Kurang lebih hasil ERD dari hasil analisis yang sebelumnya kita lakukan adalah sebagai berikut :



Gambar 9.11: ERD lengkap studi kasus Perpustakaan Sekolah

9.4 Tahapan Merubah ERD menjadi Struktur Tabel

Langkah terakhir pada perancangan basis data dari studi kasus perpustakaan sekolah ini yaitu merubah ERD lengkap menjadi struktur tabel. Struktur tabel merupakan kumpulan dari tabel-tabel yang terdapat pada database yang berfungsi menyimpan data-data yang saling berhubungan (Faiqotul Himmah; Ariasih, 2020). Hal tersebut dapat membantu dan memudahkan baik bagi perancang basis data untuk mendeskripsikan struktur tabel basis data dan dapat diterapkan ke database fisik, dengan ketentuan sebagai berikut :

1. Entitas- entitas yang terbentuk dirubah menjadi tabel-tabel
2. Atribut-atribut yang terbentuk dirubah menjadi field atau kolom

Rincian dari masing-masing tabel yang terdapat dalam perancangan basis data pada kasus perpustakaan sekolah terdiri dari nama kolom, tipe data, ukuran data, dan keterangan yang dijelaskan sebagai berikut :

9.4.1 Tabel Petugas

Tabel petugas merupakan tabel yang digunakan untuk menyimpan data-data yang berkaitan dengan data petugas selaku pustakawan disekolah yang akan menggunakan sistem. Terdapat 2 field name pada tabel data petugas yaitu NamaPetugas dan Password. NamaPetugas berperan sebagai primary key pada tabel ini. Rincian dari tabel data petugas, dapat dilihat pada tabel di bawah ini:

Tabel 9.3: Tabel Petugas

Nama Kolom	Tipe Data	Ukuran Data	Keterangan
NamaPetugas	Varchar	18	<i>Primary Key</i>
Password	Char	8	<i>Not Null</i>

9.4.2 Tabel Anggota

Tabel Anggota merupakan tabel yang digunakan untuk menyimpan data-data yang berkaitan dengan data anggota selaku siswa disekolah yang akan meminjam dan mengembalikan buku. Terdapat 4 field name pada

tbl data anggota yaitu NoAnggota. NamaAnggota, NIS, dan Kelas. NoAnggota berperan sebagai *primary key* pada tabel ini. Rincian dari tabel data petugas, dapat dilihat pada tabel di bawah ini:

Tabel 9.3: Tabel Anggota

Nama Kolom	Tipe Data	Ukuran Data	Keterangan
NoAnggota	Char	6	<i>Primary Key</i>
NamaAnggota	Varchar	25	<i>Not Null</i>
NIS	Char	10	<i>Not Null</i>
Kelas	Char	6	<i>Not Null</i>

9.4.3 Tabel Buku

Tabel Buku merupakan tabel yang digunakan untuk menyimpan data-data yang berkaitan dengan data buku yang ada diperpustakaan. Terdapat 6 field name pada tabel data buku yaitu KodeBuku, JudulBuku, Pengarang, Penerbit, Tahun dan Stock. KodeBuku berperan sebagai *primary key* pada tabel ini. Rincian dari tabel data petugas, dapat dilihat pada tabel di bawah ini:

Tabel 9.5: Tabel Buku

Nama Kolom	Tipe Data	Ukuran Data	Keterangan
KodeBuku	Char	7	<i>Primary Key</i>
JudulBuku	Varchar	20	<i>Not Null</i>
Pengarang	Varchar	18	<i>Not Null</i>
Penerbit	Varchar	18	<i>Not Null</i>
Tahun	Char	4	<i>Not Null</i>
Stock	Tinyint		<i>Not Null</i>

9.4.4 Tabel RakBuku

Tabel RakBuku merupakan tabel yang digunakan untuk menyimpan data-data yang berkaitan dengan data rak buku yang ada diperpustakaan. Terdapat 3 field name pada tabel data rak buku yaitu KodeRakBuku, KodeBuku, dan JudulBuku. KodeRakBuku berperan sebagai *primary key* pada tabel ini. Rincian dari tabel data rak buku, dapat dilihat pada tabel di bawah ini:

Tabel 9.6: Tabel RakBuku

Nama Kolom	Tipe Data	Ukuran Data	Keterangan
KodeRakBuku	Char	10	<i>Primary Key</i>
KodeBuku	Char	7	<i>Not Null</i>
JudulBuku	Varchar	20	<i>Not Null</i>

9.4.5 Tabel Peminjaman

Tabel Peminjaman merupakan tabel yang digunakan untuk menyimpan data-data yang berkaitan dengan data peminjaman buku yang ada diperpustakaan. Terdapat 5 field name pada tabel data buku yaitu NoAnggota, KodeBuku, NamaPetugas, TglPinjam dan TglKembali. NoAnggota, KodeBuku, NamaPetugas berperan sebagai *foreign key* pada tabel ini. Rincian dari tabel data peminjaman, dapat dilihat pada tabel di bawah ini:

Tabel 9.7: Tabel Buku

Nama Kolom	Tipe Data	Ukuran Data	Keterangan
NoAnggota	Char	6	<i>Foreign Key</i>
KodeBuku	Char	7	<i>Foreign Key</i>
NamaPetugas	Varchar	18	<i>Foreign Key</i>
TglPinjam	Date		<i>Not Null</i>
TglKembali	Date		<i>Not Null</i>

9.4.6 Tabel Pengembalian

Tabel Pengembalian merupakan tabel yang digunakan untuk menyimpan data-data yang berkaitan dengan data pengembalian buku yang ada diperpustakaan. Terdapat 5 field name pada tabel data buku yaitu NoAnggota, KodeBuku, NamaPetugas, TglPinjam dan TglKembali. NoAnggota, KodeBuku, NamaPetugas berperan sebagai *foreign key* pada tabel ini. Rincian dari tabel data peminjaman, dapat dilihat pada tabel di bawah ini:

Tabel 9.8: Tabel Buku

Nama Kolom	Tipe Data	Ukuran Data	Keterangan
NoAnggota	Char	6	<i>Foreign Key</i>
KodeBuku	Char	7	<i>Foreign Key</i>
NamaPetugas	Varchar	18	<i>Foreign Key</i>
TglKembali	Date		<i>Not Null</i>
DendaHari	Tinyint		<i>Not Null</i>
JumlahHari	Tinyint		<i>Not Null</i>
TotalDenda	Int		<i>Not Null</i>

Bab 10

Perancangan Basis Data Sistem Penjualan

10.1 Analisa Kebutuhan Sistem

Perancangan basis data dapat dimulai dengan penentuan kebutuhan sistem yang dibangun. Kebutuhan dapat berupa informasi maupun fitur yang dapat dihasilkan oleh sistem. Pada bab ini akan dijabarkan perancangan basis data untuk sistem informasi penjualan dimulai dari penentuan kebutuhan fitur maupun informasi yang dapat dihasilkan oleh sistem hingga perwujudan pada MariaDB.

Secara umum sistem informasi dibangun dengan memetakan 2 kebutuhan utama yaitu informasi dan fitur. Bentuk basis data maupun data yang ada di dalamnya ditentukan oleh pemetaan kebutuhan. Pemetaan kebutuhan ini tentunya sangat berpengaruh pada rancangan basis data. Bahkan ada kemungkinan bentuk basis data akan sangat berbeda walaupun basis data tersebut digunakan untuk sistem informasi sejenis. Pada contoh berikut akan dibahas beberapa pemetaan kebutuhan untuk sistem informasi penjualan dalam bentuk *relational database* (RDBMS).

10.1.1 Analisa Informasi

Data yang telah diolah disajikan dalam bentuk informasi. Informasi dapat sebagai data apabila informasi tersebut diolah kembali untuk menjadi informasi yang lain. Tujuan utama dari pembuatan sistem informasi adalah menghasilkan informasi yang dapat dipahami dan membantu pengguna sistem tersebut. Informasi dapat disajikan dalam berbagai bentuk dari yang paling sederhana sampai dengan yang lebih rumit. Informasi sederhana yang dapat disajikan yaitu dalam bentuk angka, misalkan saja jumlah total barang terjual. Sedangkan informasi yang lebih rumit dapat disajikan dalam bentuk gambar, contohnya berupa grafik penjualan. Sebelum melakukan perancangan basis data, perlu dipastikan informasi apa saja yang akan dihasilkan oleh sistem. Hal ini terkait dengan kebutuhan data yang akan diolah menjadi informasi. Dalam penyusunan basis data sistem penjualan akan dibahas beberapa kebutuhan informasi dasar yaitu informasi stok, produk, dan pelanggan.

a. Informasi stok

Informasi stok dapat diketahui dengan membuat entitas data barang yang berisi jumlah stok setiap barang. Untuk informasi tambahan seperti notifikasi barang akan habis, dapat ditambahkan atribut stok minimal. Notifikasi ini akan berguna pada saat penentuan barang yang harus dibeli untuk kebutuhan penjualan selanjutnya sehingga meminimalkan kasus stok barang kosong.

b. Informasi produk

Informasi produk terlaris dan tidak laku dapat dilakukan dengan membuat atribut jumlah penjualan pada tabel transaksi penjualan. Kalkulasi jumlah penjualan setiap barang akan menentukan status barang tersebut. Informasi ini dapat digunakan untuk menentukan barang apa saja yang perlu stok lebih banyak dan barang yang tidak layak jual.

c. Informasi pelanggan

Informasi transaksi pelanggan didapatkan dengan pengolahan riwayat pembelian oleh pelanggan. Salah satu contohnya adalah penentuan pelanggan setia. Penentuan ini dapat dilakukan dengan menambahkan atribut poin pada pelanggan atau dapat juga dengan menghitung jumlah pembelian yang dilakukan oleh pelanggan. Dalam contoh ini akan menggunakan pengolahan data pembelian oleh pelanggan. Sehingga akan ditambahkan atribut pembeli pada setiap data penjualan.

10.1.2 Analisa Fitur

Fitur sistem merupakan karakteristik khusus yang ada pada sistem perangkat lunak yang dibangun. Karakteristik dapat berupa desain, kemampuan, maupun fungsi dari sistem tersebut. Fitur merupakan salah satu elemen yang membentuk dan menghasilkan informasi. Fitur juga disusun dengan tujuan mempermudah pengguna untuk memahami dan menggunakan sistem. Dalam penyusunan sistem penjualan akan dibahas beberapa fitur yang dibutuhkan.

a. Transaksi Penjualan

Fitur transaksi adalah fitur dasar dari sistem penjualan. Tentunya dalam transaksi harus mencatat data barang dan jumlah barang yang terjual. Selain itu diperlukan data karyawan yang melayani penjualan. Beberapa data perlu ditambahkan untuk mencegah kecurangan yang dapat dilakukan oleh karyawan. Seperti penambahan total harga pada setiap transaksi sehingga dapat dilakukan pengecekan silang jika terjadi pengubahan data pada daftar barang yang terjual. Transaksi penjualan juga dapat ditambahkan atribut metode pembayaran. Metode pembayaran diisi untuk mengetahui apakah pembayaran dilakukan menggunakan uang tunai, pembayaran digital, maupun pembayaran dengan perantara bank.

b. Keuangan

Fitur laporan keuangan dapat mempercepat dan mempermudah pekerjaan karyawan terkait dengan untung-rugi, neraca, jumlah penjualan, dan berbagai hal lainnya. Untuk mendukung fitur ini diperlukan data yang lengkap mengenai jumlah penjualan, jumlah pembelian, dan sisa barang yang ada. Dengan kebutuhan tersebut maka diperlukan entitas barang masuk yang lengkap dengan harga beli. Selain data barang masuk, diperlukan juga harga jual setiap barang. Harga jual pada entitas barang mungkin saja dapat berubah sewaktu-waktu. Hal ini dapat mengakibatkan ketidaksesuaian pada fitur keuangan. Maka dari itu diperlukan atribut harga setiap barang pada setiap transaksi yang terjadi. Selain beberapa data yang telah disebutkan, diperlukan juga atribut pajak untuk setiap transaksi yang dilakukan untuk mempermudah rekapitulasi pajak yang harus dibayar oleh perusahaan.

c. Data mining

Salah satu metode *data mining* yang sering digunakan pada sistem penjualan adalah analisis keranjang belanja atau disebut juga kaidah asosiasi. Analisis keranjang belanja digunakan untuk mencari barang apa saja yang sering dibeli

bersamaan. Untuk mendukung penggalian data ini, dapat dibentuk sebuah entitas yang menampung rekapitulasi barang pada setiap transaksi yang terjadi.

d. Penyesuaian stok

Sering kali ditemukan terjadi ketidak sesuaian jumlah antara stok pada sistem dan stok pada gudang. Maka dari itu diperlukan fitur untuk menyesuaikan stok yang tercatat dalam sistem. Tentunya penyesuaian stok ini harus memiliki batasan agar tidak menjadi celah kecurangan oleh kayawan. Maka dari itu, dalam penyesuaian stok diperlukan atribut pengguna sistem yang melakukan penyesuaian stok dan keterangan jika ada selisih stok yang ditemukan.

10.2 Analisa Kebutuhan Tipe Data

Selain analisis fitur dan informasi untuk penentuan penentuan entitas dan atribut, perlu juga dilakukan penentuan tipe data dan besar data tersebut. Penentuan ini penting karena harus dilakukan kajian sebelum memutuskan tipe dan besar data pada atribut. Berikut beberapa contoh analisis tentang kebutuhan tipe data dan besar data.

a. Nama

Untuk mencegah kekurangan tempat menyimpan karakter nama, perlu dilakukan kajian tentang nama yang panjang dan unik. Misalkan saja di Indonesia ada nama “Aiwinur Siti Diah Ayu Mega Ningrum Dwi Pangestuti Lestasi Endang Pamikasih Sri Kumala Sari Dewi Puspita Anggraini” yang memiliki panjang karakter 113. Ada pula nama yang menggunakan simbol seperti petik satu (‘). Dari kasus tersebut dapat disimpulkan nama sebaiknya menggunakan tipe data variabel character dengan panjang minimal 113.

b. Nomor telepon

Nomor telepon terlihat cocok menggunakan tipe data angka seperti long integer. Akan tetapi jika diperhatikan lebih seksama, nomor telepon dimulai dengan angka 0 yang akan hilang jika menggunakan tipe data angka. Penentuan tipe data nomor telepon dapat dilakukan dengan mengkaji nomor telepon yang saat ini tersedia dan yang kemungkinan digunakan pada masa depan. Nomor telepon 14 digit sudah diperkenalkan di beberapa negara. Nomor telepon ini biasanya dimulai dengan 0 atau mengganti nomer 0 paling depan menjadi sebuah kode negara misalnya nomor telp 081 338 XXX XXX dapat juga menjadi +6281 338 XXX XXX. Dengan demikian untuk nomor

telepon dapat menggunakan tipe data variabel character untuk mengakomodasi simbol “+” dengan panjang minimal 16 karakter.

c. Atribut Kunci Utama

Penentuan bentuk kunci utama atau lebih umum dikenal dengan *primary key* sangat tergantung dengan format penandaan data pada setiap perusahaan. Pada beberapa perusahaan menggunakan *barcode* produk sebagai *primary key*. Akan tetapi ada pula yang menggunakan tanda khusus seperti menandakan tahun, nomor urut, kategori, dan lain sebagainya. Penentuan tipe data dan besarnya data untuk kata kunci harus disesuaikan dengan yang digunakan oleh perusahaan.

10.3 Rancangan Basis Data

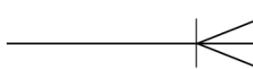
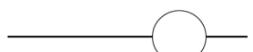
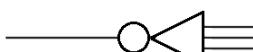
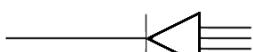
Tahapan setelah analisis kebutuhan yaitu perancangan basis data. Perancangan basis data dimulai dari penggambaran desain konsep data dapat juga disebut *Conceptual Data Model* (CDM), dilanjutkan dengan desain logika data disebut juga dengan *Logical Data Model* (LDM), dan terakhir penggambaran desain data fisik disebut juga dengan *Physical Data Model* (PDM). Penggambaran desain konsep data dan desain logika data sering kali dipilih salah satu, karena penggambaran desain logika data sudah mencakup desain konsep data. Desain ini digambarkan dengan diagram yang disebut dengan *Entity Relationship Diagram* (ERD). ERD memiliki berbagai cara penggambaran mulai dari yang paling umum dikenal dengan diagram Chen’s dan yang saat ini mulai digunakan secara luas dikenal dengan nama notasi Crow's foot. Pada bab ini akan dilakukan perancangan logika data pada basis data dengan menggunakan notasi Crow's foot.

10.3.1 Notasi Crow's Foot

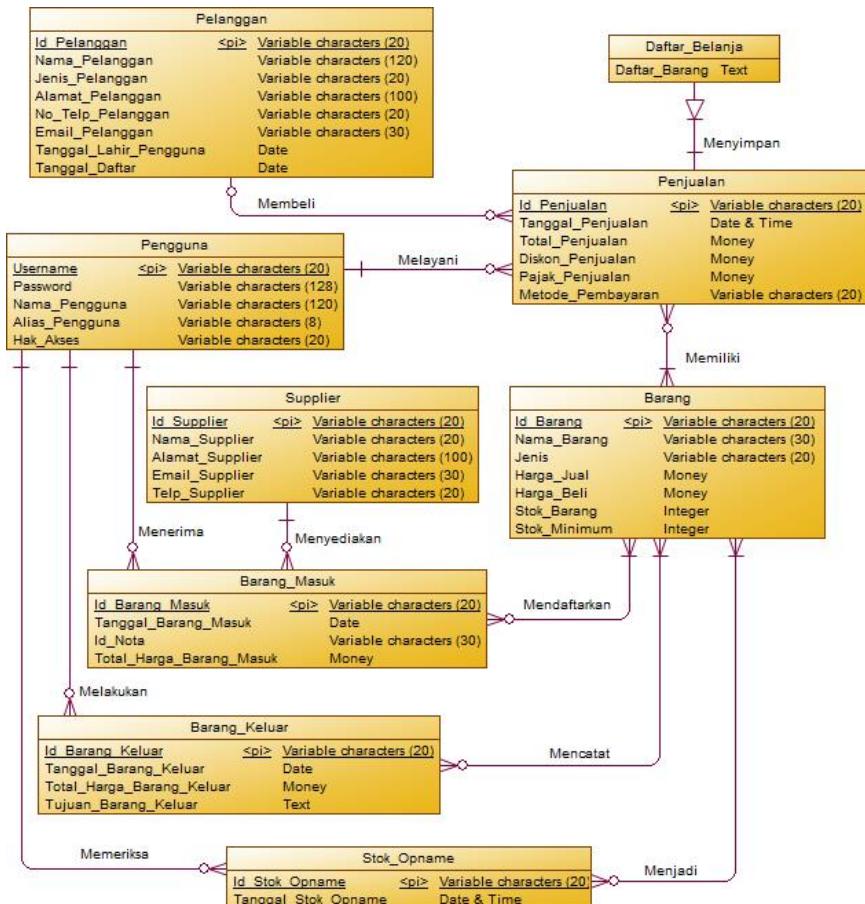
Penggambaran ERD dengan notasi Crow's foot diperkenalkan Richard Barker pada tahun 1980an, dan juga dalam pendekatan rekayasa informasi oleh James Martin dan Clive Finkelstein (Merson, 2009). Perbedaan mendasar antara notasi Chen’s dan notasi Crow's foot adalah pada bentuk notasi dan aturan notasi. Berbeda dengan notasi Chen’s, aturan pada notasi Crow's foot tidak mengijinkan sebuah relasi memiliki atribut. Relasi yang memiliki atribut pada notasi Chen’s akan menjadi sebuah entitas. Dengan demikian, penggambaran

notasi Crow's foot dapat lebih sederhana dan mudah dipahami. Penggambaran notasi Crow's foot yang lebih menyerupai tabel juga memudahkan dalam visualisasi bentuk tabel yang akan disusun. Sedangkan penggambaran relasi menggunakan garis yang memiliki ujung berbeda-beda sesuai kardinalitas dan modalitas relasi. Penggambaran notasi Crow's foot sangat tergantung dengan aplikasi visualisasi yang digunakan. Dalam visualisasi notasi Crow's foot pada contoh berikut menggunakan aplikasi PowerDesigner versi 16.6.4.3 (5517). PowerDesigner memiliki penggambaran notasi Crow's Foot ditunjukkan oleh Tabel 10.1.

Tabel 10.1: Simbol notasi Crow's Foot

No	Simbol	Nama	Keterangan
1.		Entitas (Entity)	-
2.		Wajib - Satu (Mandatory – One)	Tepat satu (Exactly One)
3.		Wajib – Banyak (Mandatory – Many)	Satu atau Banyak (One or More)
4.		Pilihan – Satu (Optional – One)	Kosong atau Satu (Zero or One)
5.		Pilihan – Banyak (Optional – Many)	Kosong atau Banyak (Zero or More)
6.		Ketergantungan (Dependent) Pilihan – Banyak (Optional – Many)	Kosong atau Banyak (Zero or More)
7.		Ketergantungan (Dependent) Wajib – Banyak (Mandatory – Many)	Satu atau Banyak (One or More)

10.3.2 Conceptual Data Model



Gambar 10.1: Visualisasi Conceptual Data Model

Penentuan entitas utama dilakukan pada saat membentuk *logical data model*. Entitas utama adalah entitas yang tidak tergantung pada entitas lainnya. Pada sistem informasi, entitas utama umumnya menyimpan data master. Entitas lain yang dibentuk adalah entitas pendukung fitur sistem informasi yang dibangun. Entitas utama pada Gambar 10.1 yaitu Pelanggan, Pengguna, Supplier, dan Barang. Entitas pendukung yaitu Penjualan, Daftar Belanja, Stok Opname, Barang Keluar, dan Barang Masuk.

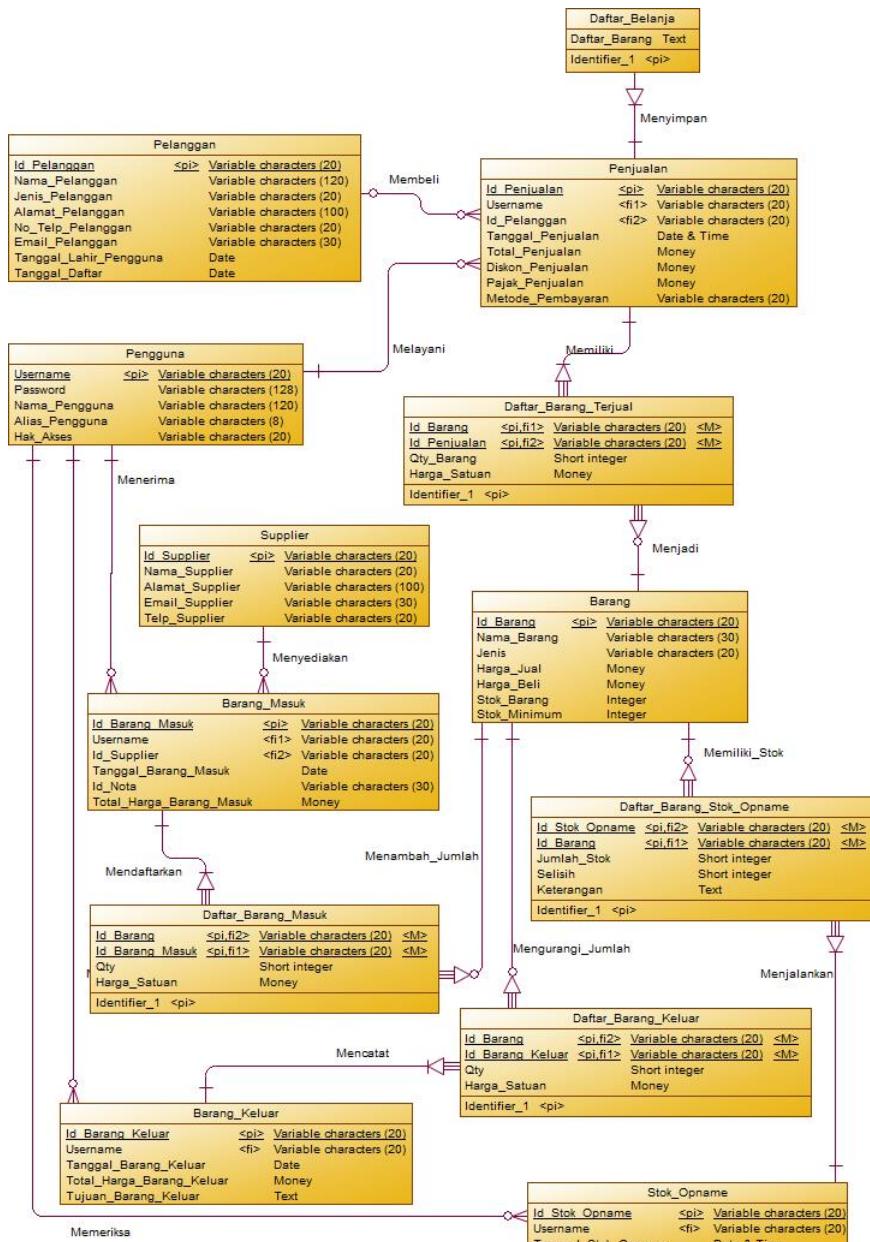
Pembuatan atribut pada setiap entitas sebisa mungkin dengan nama unik dan menyimpan data yang berbeda. Akan tetapi terkadang ada kebutuhan penyertaan atribut milik entitas lain untuk konsistensi data. Atribut harga barang adalah salah satu contohnya. Atribut ini harus dibedakan karena ada kemungkinan harga barang masuk dan harga jual barang berbeda. Selain itu ada kemungkinan juga terjadi perbedaan harga jual saat barang terjual pada waktu yang berbeda. Penyimpanan harga terjual di tempat berbeda diperlukan karena harus menyesuaikan dengan kondisi nyata pendapatan perusahaan. Sehingga laporan keuangan yang dihasilkan akan akurat sesuai dengan keadaan nyata perusahaan.

10.3.3 Logical Data Model

Entitas baru akan terbentuk saat proses pembentukan *logical data model*. Entitas ini terbentuk dari relasi *many to many*. Dalam penyusunan entitas baru pada *logical data model* dapat ditambahkan atribut-atribut yang menjadi pelengkap pada entitas tersebut. Seperti dalam Gambar 10.2 ada penambahan entitas daftar barang terjual yang terbentuk dari relasi memiliki antara entitas penjualan dan barang pada *conceptual data model*. Entitas daftar barang terjual dilengkapi oleh atribut baru seperti jumlah barang (Qty) dan harga satuan saat barang terjual. Entitas lain yang terbentuk dari relasi *many to many* yaitu daftar barang masuk, daftar barang keluar, dan daftar barang *opname*.

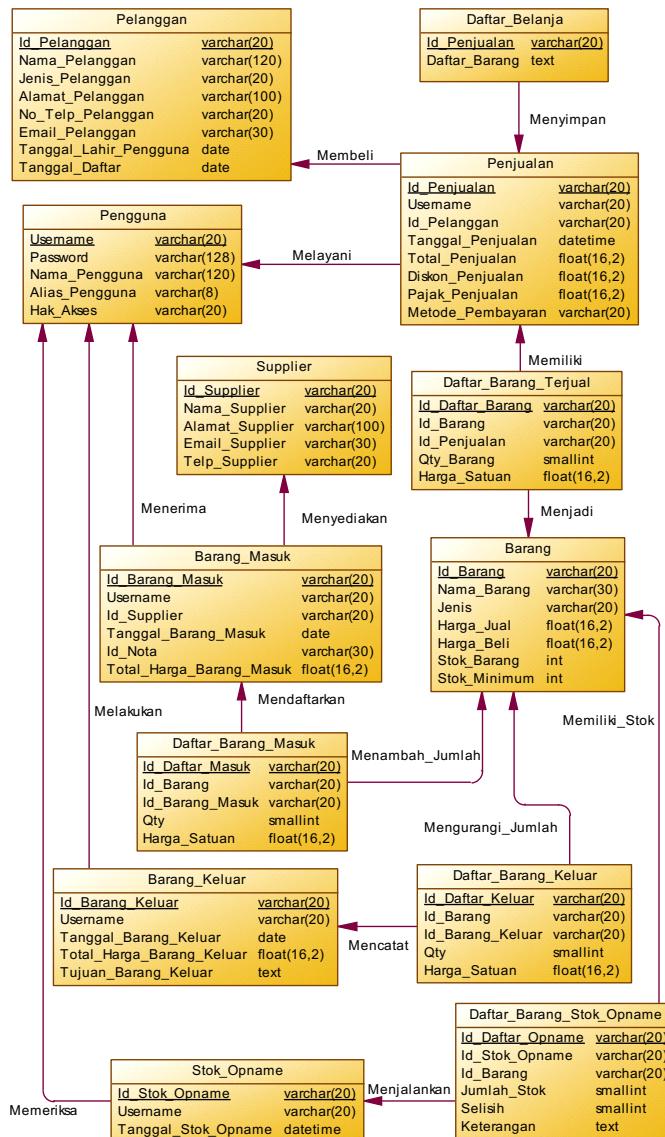
Selain entitas baru, relasi baru juga akan terbentuk dari relasi *many to many* di *logical data model*. Relasi ini umumnya memiliki sifat tergantung (*dependent*) kepada salah satu entitasnya. Dalam Gambar 10.2 ditunjukkan semua entitas yang terbentuk dari relasi *many to many* memiliki ketergantungan ke salah satu entitas yang direferensi.

Karakteristik lainnya dari *logical data model* adalah terbentuknya kunci referensi atau lebih dikenal dengan nama *foreign key*. *Foreign key* merupakan atribut yang menunjukkan bahwa suatu entitas memiliki relasi. Kunci ini muncul di ujung relasi yang bersifat *many*. Entitas yang terbentuk dari relasi *many to many* akan memiliki 2 *foreign key* yang masing-masing mereferensi entitas induk.



Gambar 10.2: Visualisasi Logical Data Model

10.3.4 Physical Data Model



Gambar 10.3: Visualisasi Physical Data Model

Foreign key pada relasi *many to many* yang membentuk entitas akan menjadi *primary key* pada entitas tersebut. Penggunaan lebih dari satu atribut untuk *primary key* terkadang akan mempersulit dan memperpanjang sintaksis MariaDB atau lebih umum dikenal dengan nama *query* MariaDB. Pembentukan *primary key* khusus dapat menjadi salah satu solusi. Pada pembentukan *physical data model* dapat ditambahkan atribut yang menjadi *primary key* khusus sehingga *foreign key* tidak lagi bertindak sebagai *primary key*. Pada Gambar 10.3 dapat dilihat atribut yang khusus ditambahkan sebagai *primary key* dalam bentuk Id setiap entitas. Dalam *physical data model* juga terjadi perubahan tipe data menyesuaikan dengan tipe data yang digunakan pada *database management system* yang digunakan.

10.4 Implementasi Rancangan Basis Data

Implementasi merupakan tahap perwujudan rancangan basis data pada DBMS yang telah ditentukan. Rancangan basis data harus melalui berbagai pertimbangan dan tahapan sebelum implementasi. Dalam penyusunan *relational database*, normalisasi dan fragmentasi merupakan salah satu tahap pada perancangan basis data. Tahap ini umumnya dilakukan untuk meningkatkan performa basis data. Akan tetapi, seringkali tahap ini tidak dilaksanakan karena basis data yang dirancang telah sesuai dengan kebutuhan. Dalam perancangan basis data penjualan yang dibahas tidak sampai normalisasi dan fragmentasi. Implementasi mencakup sintaksis pembuatan basis data dan sekaligus pembuatan atribut referensi pada MariaDB.

10.4.1 Implementasi pada MariaDB

Pilihan fasilitas hosting sangat mempengaruhi harga yang harus dibayarkan. Untuk developer yang baru belajar biasanya menggunakan *hosting* yang berbiaya rendah hingga gratis. Salah satu fasilitas yang mempengaruhi harga adalah *database management system* (DBMS) yang digunakan. MySQL merupakan DBMS yang sangat populer. Salah satu keuntungan penggunaan MySQL adalah sangat umum tersedia pada penyedia *hosting* di Indonesia dari yang gratis hingga yang berbiaya sangat tinggi. Seiring dengan waktu, MySQL mulai dikembangkan menjadi DBMS yang memiliki arah yang berbeda.

MariaDB adalah salah satu DBMS yang dikembangkan berdasarkan MySQL. Hampir semua sintaksis yang ada pada MySQL versi 5.7 juga telah tersedia pada MariaDB. Keuntungan penggunaan MariaDB adalah berbasis *opensource*, dikembangkan komunitas, dan juga telah terintegrasi pada XAMPP mulai dari versi 5.5.30 dan 5.6.14. Saat ini, penyedia *hosting* berbiaya rendah juga mulai beralih ke MariaDB. Perbedaan versi MariaDB tentunya akan mempengaruhi sintaksis yang dapat dijalankan. Pada contoh implementasi basis data berikut akan menggunakan MariaDB versi 10.4.6.

a. Implementasi Tabel

Pembuatan tabel dapat menggunakan 2 pendekatan yaitu pembuatan tabel satu persatu secara berurutan sesuai dengan jenis entitas, atau dapat pula dengan pembuatan tabel sekaligus. Pembuatan tabel dengan berurutan sesuai jenis entitas perlu memperhatikan urutan entitas utama, pendukung, dan juga entitas yang terbentuk dari relasi *many to many*. Hal ini dilakukan untuk mencegah adanya pembuatan *foreign key* yang tidak memiliki tabel referensi. Untuk mempermudah pembuatan tabel, dapat menggunakan pendekatan kedua yaitu membuat tabel sekaligus. Pada pendekatan ini, pembuatan seluruh tabel beserta atributnya dilakukan dengan mengabaikan *foreign key*. Pembuatan *foreign key* dilakukan pada saat semua tabel telah tersedia. Setiap tabel lalu diubah untuk penambahan *foreign key* dengan sintaks “ALTER”, sehingga semua *foreign key* pasti memiliki tabel yang direferensi. Pada contoh sintaksis akan menggunakan pendekatan kedua.

Tabel BARANG

```
create table BARANG
(
    ID_BARANG      varchar(20) not null comment '',
    NAMA_BARANG    varchar(30)  comment '',
    JENIS          varchar(20)  comment '',
    HARGA_JUAL     float(16,2) comment '',
    HARGA_BELI     float(16,2) comment '',
    STOK_BARANG    int        comment '',
    STOK_MINIMUM   int        comment '',
    primary key (ID_BARANG)
);
```

Tabel BARANG_KELUAR

```
create table BARANG_KELUAR
(
    ID_BARANG_KELUAR      varchar(20) not null comment '',
    USERNAME                varchar(20) not null comment '',
    TANGGAL_BARANG_KELUAR date comment '',
    TOTAL_HARGA_BARANG_MASUK float(16,2) comment '',
    TUJUAN_BARANG_KELUAR text comment '',
    primary key (ID_BARANG_KELUAR)
);
```

Tabel BARANG_MASUK

```
create table BARANG_MASUK
(
    ID_BARANG_MASUK      varchar(20) not null comment '',
    USERNAME                varchar(20) not null comment '',
    ID_SUPPLIER             varchar(20) not null comment '',
    TANGGAL_BARANG_MASUK date comment '',
    ID_NOTA                  varchar(30) comment '',
    TOTAL_HARGA_BARANG_MASUK float(16,2) comment '',
    primary key (ID_BARANG_MASUK)
);
```

Tabel DAFTAR_BARANG_KELUAR

```
create table DAFTAR_BARANG_KELUAR
(
    ID_DAFTAR_KELUAR      varchar(20) not null comment '',
    ID_BARANG                varchar(20) not null comment '',
    ID_BARANG_KELUAR        varchar(20) not null comment '',
    QTY                      smallint comment '',
    HARGA_SATUAN            float(16,2) comment '',
    primary key (ID_DAFTAR_KELUAR)
);
```

Tabel DAFTAR_BARANG_MASUK

```
create table DAFTAR_BARANG_MASUK
(
    ID_DAFTAR_MASUK      varchar(20) not null comment '',
    ID_BARANG                varchar(20) not null comment '',
    ID_BARANG_MASUK        varchar(20) not null comment '',
    primary key (ID_DAFTAR_MASUK)
);
```

```
        QTY          smallint comment '',
HARGA_SATUAN      float(16,2) comment '',
primary key (ID_DAFTAR_MASUK)
);
```

Tabel DAFTAR_BARANG_STOK_OPNAME

```
create table DAFTAR_BARANG_STOK_OPNAME
(
    ID_DAFTAR_OPNAME      varchar(20) not null comment '',
    ID_STOK_OPNAME        varchar(20) not null comment '',
    ID_BARANG              varchar(20) not null comment '',
    JUMLAH_STOK            smallint comment '',
    SELISIH                smallint comment '',
    KETERANGAN             text comment '',
    primary key (ID_DAFTAR_OPNAME)
);
```

Tabel DAFTAR_BARANG_TERJUAL

```
create table DAFTAR_BARANG_TERJUAL
(
    ID_DAFTAR_BARANG      varchar(20) not null comment '',
    ID_BARANG              varchar(20) not null comment '',
    ID_PENJUALAN           varchar(20) not null comment '',
    QTY_BARANG              smallint comment '',
    HARGA_SATUAN           float(16,2) comment '',
    primary key (ID_DAFTAR_BARANG)
);
```

Tabel DAFTAR_BELANJA

```
create table DAFTAR_BELANJA
(
    ID_PENJUALAN           varchar(20) not null comment '',
    DAFTAR_BARANG           text comment '',
    primary key (ID_PENJUALAN)
);
```

Tabel PELANGGAN

```
create table PELANGGAN
(
    ID_PELANGGAN      varchar(20) not null comment '',
    NAMA_PELANGGAN    varchar(120) comment '',
    JENIS_PELANGGAN   varchar(20) comment '',
    ALAMAT_PELANGGAN  varchar(100) comment '',
    NO_TELP_PELANGGAN varchar(20) comment '',
    EMAIL_PELANGGAN   varchar(30) comment '',
    TANGGAL_LAHIR_PENGGUNA date comment '',
    TANGGAL_DAFTAR     date comment '',
    primary key (ID_PELANGGAN)
);
```

Tabel PENGGUNA

```
create table PENGGUNA
(
    USERNAME           varchar(20) not null comment '',
    PASSWORD           varchar(128) comment '',
    NAMA_PENGGUNA     varchar(120) comment '',
    ALIAS_PENGGUNA    varchar(8) comment '',
    HAK_AKSES          varchar(20) comment '',
    primary key (USERNAME)
);
```

Tabel PENJUALAN

```
create table PENJUALAN
(
    ID_PENJUALAN      varchar(20) not null comment '',
    USERNAME           varchar(20) not null comment '',
    ID_PELANGGAN       varchar(20) comment '',
    TANGGAL_PENJUALAN  datetime comment '',
    TOTAL_PENJUALAN    float(16,2) comment '',
    DISKON_PENJUALAN   float(16,2) comment '',
    PAJAK_PENJUALAN    float(16,2) comment '',
    METODE_PEMBAYARAN  varchar(20) comment '',
    primary key (ID_PENJUALAN)
);
```

Tabel STOK_OPNAME

```
create table STOK_OPNAME
(
    ID_STOK_OPNAME      varchar(20) not null comment '',
    USERNAME             varchar(20) not null comment '',
    TANGGAL_STOK_OPNAME datetime comment '',
    primary key (ID_STOK_OPNAME)
);
```

Tabel SUPPLIER

```
create table SUPPLIER
(
    ID_SUPPLIER          varchar(20) not null comment '',
    NAMA_SUPPLIER        varchar(20) comment '',
    ALAMAT_SUPPLIER      varchar(100) comment '',
    EMAIL_SUPPLIER        varchar(30) comment '',
    TELP_SUPPLIER         varchar(20) comment '',
    primary key (ID_SUPPLIER)
);
```

b. Implementasi *foreign key*

Foreign key merupakan kolom yang digunakan sebagai penghubung antar tabel. Pembuatan *foreign key* memanfaatkan sintaksis “ALTER”. Struktur sintaksis yaitu ALTER TABLE “nama_tabel” ADD CONSTRAINT “nama_relati” FOREIGN KEY (“nama_kolom”) REFERENCES “tabel_referensi” (“kolom_referensi”) ON DELETE RESTRICT ON UPDATE RESTRICT. DELETE RESTRICT menyatakan bahwa penghapusan data pada tabel referensi (*parent*) akan dibatasi jika masih ada data yang mereferensi di tabel lain (*child*). Sedangkan UPDATE RESTRICT menyatakan pembatasan perubahan data yang direferensi. Berikut merupakan sintaksis yang dikelompokkan berdasarkan tabel yang direferensi

Referensi PENGGUNA

```
ALTER TABLE BARANG_KELUAR ADD CONSTRAINT
FK_BARANG_K_MELAKUKAN_PENGGUNA FOREIGN KEY (USERNAME)
    REFERENCES PENGGUNA (USERNAME) ON DELETE RESTRICT ON
    UPDATE RESTRICT;
```

```
ALTER TABLE BARANG_MASUK ADD CONSTRAINT  
FK_BARANG_M_MENERIMA_PENGGUNA FOREIGN KEY (USERNAME)  
    REFERENCES PENGGUNA (USERNAME) ON DELETE RESTRICT ON  
UPDATE RESTRICT;
```

```
ALTER TABLE PENJUALAN ADD CONSTRAINT  
FK_PENJUALA_MELAYANI_PENGGUNA FOREIGN KEY (USERNAME)  
    REFERENCES PENGGUNA (USERNAME) ON DELETE RESTRICT ON  
UPDATE RESTRICT;
```

```
ALTER TABLE STOK_OPNAME ADD CONSTRAINT  
FK_STOK_OPN_MEMERIKSA_PENGGUNA FOREIGN KEY (USERNAME)  
    REFERENCES PENGGUNA (USERNAME) ON DELETE RESTRICT ON  
UPDATE RESTRICT;
```

Referensi BARANG

```
ALTER TABLE DAFTAR_BARANG_KELUAR ADD CONSTRAINT  
FK_DAFTAR_B_MENGURANG_BARANG FOREIGN KEY (ID_BARANG)  
    REFERENCES BARANG (ID_BARANG) ON DELETE RESTRICT ON  
UPDATE RESTRICT;
```

```
ALTER TABLE DAFTAR_BARANG_MASUK ADD CONSTRAINT  
FK_DAFTAR_B_MENAMBAH_BARANG FOREIGN KEY (ID_BARANG)  
    REFERENCES BARANG (ID_BARANG) ON DELETE RESTRICT ON  
UPDATE RESTRICT;
```

```
ALTER TABLE DAFTAR_BARANG_STOK_OPNAME ADD CONSTRAINT  
FK_DAFTAR_B_MEMILIKI_BARANG FOREIGN KEY (ID_BARANG)  
    REFERENCES BARANG (ID_BARANG) ON DELETE RESTRICT ON  
UPDATE RESTRICT;
```

```
ALTER TABLE DAFTAR_BARANG_TERJUAL ADD CONSTRAINT  
FK_DAFTAR_B_MENJADI_BARANG FOREIGN KEY (ID_BARANG)  
    REFERENCES BARANG (ID_BARANG) ON DELETE RESTRICT ON  
UPDATE RESTRICT;
```

Referensi PENJUALAN

```
ALTER TABLE DAFTAR_BARANG_TERJUAL ADD CONSTRAINT  
FK_DAFTAR_B_MEMILIKI_PENJUALA FOREIGN KEY (ID_PENJUALAN)  
    REFERENCES PENJUALAN (ID_PENJUALAN) ON DELETE RESTRICT  
ON UPDATE RESTRICT;
```

```
ALTER TABLE DAFTAR_BELANJA ADD CONSTRAINT  
FK_DAFTAR_B_MENYIMPAN_PENJUALA FOREIGN KEY (ID_PENJUALAN)  
    REFERENCES PENJUALAN (ID_PENJUALAN) ON DELETE RESTRICT  
ON UPDATE RESTRICT;
```

Referensi BARANG_MASUK

```
ALTER TABLE DAFTAR_BARANG_MASUK ADD CONSTRAINT  
FK_DAFTAR_B_MENDAFTAR_BARANG_M FOREIGN KEY (ID_BARANG_MASUK)  
    REFERENCES BARANG_MASUK (ID_BARANG_MASUK) ON DELETE  
RESTRICT ON UPDATE RESTRICT;
```

Referensi BARANG_KELUAR

```
ALTER TABLE DAFTAR_BARANG_KELUAR ADD CONSTRAINT  
FK_DAFTAR_B_MENCATAT_BARANG_K FOREIGN KEY (ID_BARANG_KELUAR)  
    REFERENCES BARANG_KELUAR (ID_BARANG_KELUAR) ON DELETE  
RESTRICT ON UPDATE RESTRICT;
```

Referensi STOK_OPNAME

```
ALTER TABLE DAFTAR_BARANG_STOK_OPNAME ADD CONSTRAINT  
FK_DAFTAR_B_MENJALANK_STOK_OPN FOREIGN KEY (ID_STOK_OPNAME)  
    REFERENCES STOK_OPNAME (ID_STOK_OPNAME) ON DELETE  
RESTRICT ON UPDATE RESTRICT;
```

Referensi PELANGGAN

```
ALTER TABLE PENJUALAN ADD CONSTRAINT  
FK_PENJUALA_MEMBELI_PELANGGA FOREIGN KEY (ID_PELANGGAN)  
    REFERENCES PELANGGAN (ID_PELANGGAN) ON DELETE RESTRICT  
ON UPDATE RESTRICT;
```

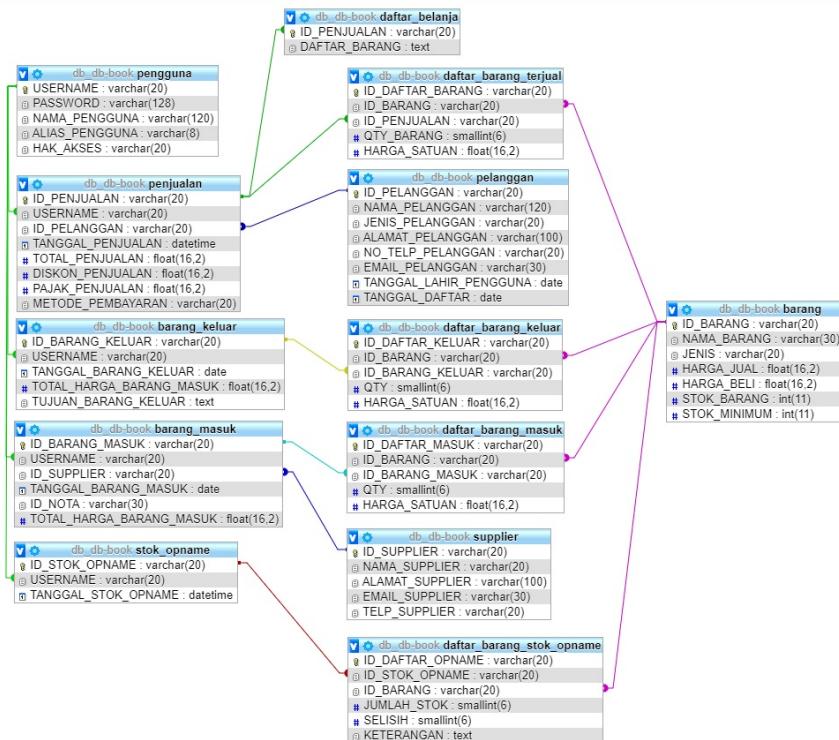
Referensi SUPPLIER

```
ALTER TABLE BARANG_MASUK ADD CONSTRAINT  
FK_BARANG_M_MENYEDIAK_SUPPLIER FOREIGN KEY (ID_SUPPLIER)  
    REFERENCES SUPPLIER (ID_SUPPLIER) ON DELETE RESTRICT ON  
UPDATE RESTRICT;
```

10.4.2 Struktur Tabel

Hasil eksekusi sintaksis pada MariaDB akan membentuk struktur tabel pada Gambar 10.4. Bentuk visualisasi struktur tabel dalam bentuk *relational view*

mirip dengan *physical data model*. Perbedaan utamanya ada pada garis yang menghubungkan setiap tabel. Pada visualisasi struktur tabel, tanda garis menunjukkan kolom yang direferensi. Visualisasi pada Gambar 10.4 merupakan visualisasi yang dihasilkan dari phpMyAdmin versi 4.9.0.1. Setiap tabel direpresentasikan dengan bentuk persegi dilengkapi nama tabel, simbol keterangan kunci, nama kolom, dan tipe data.



Gambar 10.4: Visualisasi struktur tabel



Daftar Pustaka

A.S., Rosa dan Shalahudin, M. (2013) Rekayasa Perangkat Lunak Terstruktur dan Berorientasi objek. Bandung: Informatika.

Adi, S. and Kristin, D. M. (2014) ‘Strukturisasi Entity Relationship Diagram dan Data Flow Diagram Berbasis Business Event-Driven’, ComTech: Computer, Mathematics and Engineering Applications, 5(1). doi: 10.21512/comtech.v5i1.2577.

Amazon. (2020). Apa itu Database Relasional? Retrieved August 30, 2020, from <https://aws.amazon.com/id/relational-database/>

Arsyad, L. and Soeratno, D., (1993). Metode Penelitian Untuk Ekonomi dan Bisnis.

Brady, M. and Loonam, J. (2010) ‘Exploring the use of entity□relationship diagramming as a technique to support grounded theory inquiry’, Qualitative Research in Organizations and Management: An International Journal, 5(3), pp. 224–237. doi: 10.1108/17465641011089854.

Connolly, T. and Begg, C. (2005) ‘Database Systems’, in Fourth Edition. Fourth. Addison Wesley, p. 342. Available at: www.booksites.net/connbegg.

Connoly, T. M., Begg, C. E. and Strachan, A. D. (1996) Database System. A Practical Approach to Design, Implementation, and Management. Addison Wesley Company.

Coronel, C. and Morris, S. (2016) Database systems: design, implementation, & management. Cengage Learning.

Coronel, C. and Morris, S. (2016) Database Systems: Design, Implementation, and Management.

- Coronel, C. and Morris, S. (2019) Database System Design, Implementation & Management. 13th Editi.
- Coronel, C., & Morris, S. (2019). Database systems : design, implementation, and management (13th ed.). Australia: Cengage Learning.
- dan Supomo, I., (2012). Metodologi Penelitian Untuk Akutansi dan Manajemen. Cetakan kedua, Yogyakarta: BPFE.
- Davis, G.B., (1999). Sistem Informasi Manajemen. Penerbit Andi, Yogyakarta.
- Faiqotul Himmah;Ariasih (2020) Rancang Bangun Sistem Peramalan Pajak Berbasis Website dengan Metode Single Exponential Smoothing Pada CV. JAya Utama Mandiri Solution.
- Geeksforgeeks (2019) Categories of End Users in DBMS. Available at: <https://www.geeksforgeeks.org/categories-of-end-users-in-dbms/> (Accessed: 1 October 2020).
- Gupta, S. B., & Mittal, A. (2017). Introduction to Database Management System, 2nd Edition. Delhi: University Science Press.
- Hargo, A. (2006) ‘Normalisasi Database’, pp. 1–9.
- Harrington, J. L. (2016) Relational database design and implementation. Morgan Kaufmann.
- Hutahaean, J. (2015) Konsep sistem informasi. Deepublish.
- IBM. (2020a). Relational Database. Retrieved August 30, 2020, from <https://www.ibm.com/ibm/history/ibm100/us/en/icons/reldb/>
- IBM. (2020b). What is a relational database? Retrieved August 30, 2020, from <https://www.ibm.com/analytics/relational-database>
- Indrajani, S. (2017) Database Design... Theory, Practice, and Case Study. Elex Media Komputindo.
- Irmansyah, F. (2003) ‘Pengantar Database’, Pengantar Database, pp. 1–13. Available at: <https://staff.blog.ui.ac.id/r-suti/files/2010/03/pengantardb.pdf>.
- Iyer, V. K. (2015) ‘Functional dependency in relational databases (adapted after M.Y.Vardi’s survey)’, (March 2011).

- James V Luisi (2014) Pragmatic Enterprise Architecture: Strategies to Transform Information Systems in the Era of Big Data. United States of America: Elsevier Inc. Available at: <https://www.sciencedirect.com/book/9780128002056/pragmatic-enterprise-architecture>.
- Kadir, A., (2009). Dasar perancangan dan implementasi database relasional. Yogyakarta: Andi.
- Kadir.A (2008) Dasar Perancangan & Implementasi Database Relasional. Yogyakarta: Andi Offset.
- Korth, H. F. and Silberschatz, A. (2010) 'Database system concepts 6th Edition', Chapter, 11, p. 510.
- Kroenke, M. D. (2003) Database Processing. Jakarta: Erlangga.
- Kusrini, 2007. Strategi Perancangan dan Pengelolaan Basis Data. First Edition ed. Yogyakarta: C.V ANDI OFFSET (Penerbit ANDI).
- Kusrini, M. K. (2007) Konsep Dan Aplikasi Sistem Pendukung Keputusan, Penerbit Andi.
- Larassati, M. et al. (2019) 'Pengembangan Sistem Pemetaan Otomatis Entity Relationship Diagram Ke Dalam Database', Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, 3(4), pp. 4058–4065. Available at: <http://j-ptiik.ub.ac.id>.
- Lemahieu, W., Broucke, S. vanden, & Baesens, B. (2018). Principles of database management: the practical guide to storing, managing and analyzing big and small data. Cambridge, United Kingdom: Cambridge University Press.
- MariaDB (2015a) Understanding the Hierarchical Database Model. Available at: <https://mariadb.com/kb/en/understanding-the-network-database-model/> (Accessed: 5 October 2020).
- MariaDB (2015b) Understanding the Network Database Model. Available at: <https://mariadb.com/kb/en/understanding-the-network-database-model/> (Accessed: 5 October 2020).
- Merson, P. (2009) 'Data Model as an Architectural View', Solutions.

- Naik, S. (2014). Concept Of Database Management System. New Delhi, India: Pearson.
- Nugroho, A. (2011) Perancangan dan Implementasi Sistem Basis Data. Edited by Nikodemus WK. Yogyakarta: Penerbit Andi Yogyakarta.
- Nur Hidayat (2016) Membuat Model Basis Data, pojok programmer. Available at: https://pojokprogrammer.net/id/content/membuat-model-basis-data?language_content_entity=id.
- Oracle. (2020). What Is a Relational Database? Retrieved August 30, 2020, from <https://www.oracle.com/database/what-is-a-relational-database/>
- Puspitasari, D., Rahmad, C. and Astiningrum, M. (2016) ‘Normalisasi Tabel Pada Basisdata Relasional’, Prosiding SENTIA, 8(1), pp. 340–345.
- Rainer, R.K., Prince, B., Splettstoesser-Hogeterp, I., Sanchez-Rodriguez, C. and Ebrahimi, S., (2020). Introduction to information systems. John Wiley & Sons.
- Salmubi, S. (2020) ‘Program Literasi Informasi: Sebuah Upaya Pemberdayaan Pemakai Perpustakaan dalam Mewujudkan Pendidikan Bermutu’, Media Pustakawan, 14(3&4), pp. 135–145.
- Setiadi, M. F. (2017) ‘E-Book Sistem Database’, Archipel, 13(1), pp. 15–20. doi: 10.3406/arch.1977.1322.
- Setyarini, I. (2016) ‘Perancangan Sistem Informasi Posyandu Guna Mendukung Pelaporan Data Perkembangan Bayi Dan Balita’, Artikel Skripsi, Universitas Nusantara PGRI Kediri, 1(1), pp. 1–11. Available at: http://simki.unpkediri.ac.id/mahasiswa/file_artikel/2016/12.1.03.03.0247.pdf.
- Sherman, R. (2015) Business Intelligence Guidebook : From Data Integration to Analytics. United States of America: Elsevier Inc. Available at: <https://www.sciencedirect.com/science/article/pii/B9780123751065000038>.
- Silberschatz, A., Korth, H. F. and Sudarshan, S. (1997) Database system concepts. McGraw-Hill New York.
- Silberschatz, A., Korth, H. F. and Sudarshan, S. (2011) Database System Concepts - 6th. ed., Database. doi: 10.1145/253671.253760.

- Silberschatz, A., Korth, H. F. and Sudarshan, S. (2011) Database System Concepts. 6th Editio. The McGrawHill Companies.
- Silberschatz, A., Korth, H. F. and Sudarshan, S. (2019) Database System Concepts (7th. edition), McGraw Hill. doi: 10.1145/253671.253760.
- Silberschatz, A., Korth, H. F. and Sudarshan, S. (2019) Database System Concepts (7th. edition), McGraw Hill. doi: 10.1145/253671.253760.
- Silberschatz, A., Korth, H., & Sudarshan, S. (2020). Database system concepts. New York, NY, USA: McGraw-Hill Education.
- Simarmata, J. and Paryudi, I., (2006). Basis data. Yogyakarta: Andi Offset.
- Simarmata, J., 2007. Perancangan Basis Data. First Edition ed. Yogyakarta: C.V ANDI OFFSET.
- Siyoto, S. and Sodik, M.A.,(2015). Dasar metodologi penelitian. Literasi Media Publishing.
- Sugiyono, P.D., (2017). Metode Penelitian Bisnis: Pendekatan Kuantitatif, Kualitatif, Kombinasi, dan R&D. Penerbit CV. Alfabeta: Bandung.
- Suryana, D., 2012. Sistem Teknologi Informasi Jilid 3: Sistem Informasi Penggajian Karyawan. First Edition ed. Scotts Valley: CreateSpace Independent Publishing Platform.
- Sutabri, T., (2012). Analisis sistem informasi. Penerbit Andi.
- Sutanta, E., (2004). Sistem Basis Data. Yogyakarta: Graha Ilmu.
- Teorey, T. J. et al. (2011) Database modeling and design: logical design. Elsevier.
- Thalheim, B. (2013) Dependencies in Relational Databases. doi: 10.1007/978-3-663-12018-6.
- Wati, I. R. (2013) ‘Analisis dan Perancangan Basis Data Perpustakaan (Studi Kasus Pada SMK Panggali Nusantara Palembang)’, Jurnal Teknologi, 1(1), pp. 69–73. doi: 10.11113/jt.v56.60.
- Watt, A. (2014) Database Design - 2nd Edition. Creative Commons. Available at: <https://opentextbc.ca/dbdesign01/>.

West, M. (2011) Developing High Quality Data Models, Developing High Quality Data Models. London: Shell International Limited. doi: 10.1016/c2009-0-30508-5.

Yusuf, A. Muri. (2014). Metode Penelitian Kuantitatif, Kualitatif & Penelitian Gabungan. Jakarta: Kencana.

Biodata Penulis



Ni Luh Wiwik Sri Rahayu Ginantra

Lahir dan besar di Bali. Pendidikan TK hingga SD diselesaikan di Kabupaten Bangli dan Pendidikan SMP hingga SMA di Kota Denpasar. Menyelesaikan Pendidikan S1 Teknik Informatika di Institut Teknologi Adhi Tama Surabaya dan S2 Ilmu Komputer Universitas Pendidikan Ganesha Singaraja. Dosen tetap di STMIK STIKOM Indonesia (STIKI Indonesia). Mengajar mata kuliah Analisa Desain Sistem Informasi, Kecerdasan Buatan, Enterprise

Information System, Software Engineering, Human Computer Interaction, Database.



Ni Wayan Wardani, S.Kom., M.Kom

Menyelesaikan S1 di Universitas Kristen Duta Wacana Yogyakarta Jurusan Teknik Informatika dan S2 di Universitas Pendidikan Ganesha Jurusan Ilmu Komputer. Saat ini adalah Dosen tetap Program studi Teknik Informatika di STMIK STIKOM Indonesia Bali. Mengampu mata kuliah Basis Data, Algoritma dan Pemrograman, dan Sistem Pendukung Keputusan. Buku yang telah ditulis dan diterbitkan berjudul *Penerapan Data Mining dalam Analytic CRM*, selain itu aktif menulis artikel di berbagai jurnal ilmiah nasional dan internasional. Salah satu artikel yang berjudul *Comparison Of*

Fuzzy Mamdani And Fuzzy Tsukamoto's Methods In Rice Selection dimuat di Jurnal Internasional bereputasi terindex Scopus.



I Gusti Ayu Agung Mas Aristamy, S.TI., M.Kom.

Menempuh pendidikan TK hingga SMA di Sekolah Kristen Harapan Denpasar. Menyelesaikan pendidikan S1 Teknologi Informasi di Universitas Udayana Bali, lalu dilanjutkan dengan menyelesaikan pendidikan S2 di Institut Teknologi Sepuluh Nopember (ITS) Surabaya. Saat ini menjadi Dosen tetap di STMIK

STIKOM Indonesia (STIKI Indonesia) di kota Denpasar. Matakuliah yang diajar meliputi Algoritma Pemrograman, Enterprise Information System, Decision Support System, Human Computer Interaction, Bisnis Berbasis Teknologi.



I Wayan Dharma Suryawan, S.Kom., M.Cs.

Lahir di Denpasar serta menghabiskan masa kecil hingga remaja di kota Denpasar. Pendidikan S1 ditempuh di Program Studi S1 Ilmu Komputer Universitas Gadjah Mada dan pendidikan S2 ditempuh di program S2 Ilmu Komputer Universitas Gadjah Mada. Saat ini aktif mengajar di STMIK STIKOM Indonesia (STIKI Indonesia) di kota Denpasar. Mata kuliah yang diampu terdiri dari Algoritma dan Pemrograman, Struktur Data, *Object Oriented Programming*, *Mobile Programming*.

**Dewa Putu Yudhi Ardiana, S.Kom., M.Pd**

Penulis lahir di ujung barat pulau bali pada tahun 1987. Merupakan anak tertua dari dua bersaudara. Mempunyai hobi membaca dan sepakbola. Penulis mempunyai prinsip bahwa kegagalan adalah awal dari kesuksesan sehingga jangan pernah menyerah untuk mencoba. Penulis berlatar belakang pendidikan sarjana Teknik Informatika dan magister Teknologi Pembelajaran. Penulis aktif sebagai dosen di STMIK STIKOM Indonesia dengan mata kuliah yang diampu berkaitan dengan programming, human computer interaction, e-commerce dan gamifikasi.

**I Gede Iwan Sudipa, S.Kom., M.Cs.**

Lahir di Singaraja, Bali. Pendidikan TK hingga SMP diselesaikan di Kabupaten Singaraja dan melanjutkan Pendidikan SMA di Denpasar. Menempuh Pendidikan S1 Teknik Informatika di STMIK AKAKOM Yogyakarta dan melanjutkan Pendidikan S2 Ilmu Komputer Universitas Gadjah Mada Yogyakarta. Saat ini sebagai Dosen tetap di STMIK STIKOM Indonesia (STIKI Indonesia) di kota Denpasar. Mengajar mata kuliah Sistem Pendukung Keputusan, Basis Data, Pemrograman Web dan Analisa Desain dan Sistem Informasi.

**Ayu Manik Dirgayusari, S.Kom., M.MT**

Lahir dan besar di Denpasar, Bali. Pendidikan TK hingga SMA diselesaikan di Kota Denpasar. Menyelesaikan Pendidikan S1 Sistem Komputer di STIKOM Surabaya yang sekarang berganti nama menjadi Universitas Dinamika dan S2 Manajemen Teknologi Informasi di Institut Teknologi Sepuluh Nopember (ITS) di Surabaya. Sebagai Dosen di STMIK STIKOM Indonesia (STIKI Indonesia) di kota Denpasar. Mengajar mata kuliah Algoritma dan Pemrograman, Human Computer Interaction, Database.

**Gede Surya Mahendra, S.Pd., M.Kom**

Lahir di Singaraja, 13 Maret 1990, dan telah menyelesaikan pendidikan S1 Jurusan Pendidikan Teknik Informatika di Universitas Pendidikan Ganesha dan melanjutkan di Program Pascasarjana Program Studi Ilmu Komputer di Universitas Pendidikan Ganesha dengan spesialisasi di bidang Decision Support System. Penulis merupakan Dosen pada STIMIK STIKOM Indonesia sejak 2018. Mata Kuliah yang pernah diampu diantaranya Decision Support System, Bisnis Berbasis Teknologi, Analisis dan Desain SI, Artificial Intelligence, Teknologi Informasi Pariwisata, Human Computer Interaction, Audit TI dan Tata Kelola TI.

**Ni Kadek Ariasih, S.Kom,M.T.**

Lahir dan besar di Bali. Pendidikan TK hingga SMA di Kota Denpasar. Menyelesaikan Pendidikan S1 Sistem Informasi di Universitas Dinamika Surabaya dan S2 Teknik Elektro di Universitas Udayana. Dosen di STMIK STIKOM Indonesia (STIKI Indonesia) di kota Denpasar. Mengajar mata kuliah Analisa Desain Sistem Informasi, Object Oriented Design Analysis, Pengantar Teknologi Informasi, Sistem Informasi Manajemen, Software Engineering, Human Computer Interaction, Basis Data, Data Integration, Object Oriented Programming.

**Wayan Gede Suka Parwita, M.Cs.**

Lahir dan besar di Bali. Pendidikan SD diselesaikan di Kabupaten Badung dan Pendidikan TK, SMP, hingga SMK di Kota Denpasar. Menyelesaikan Pendidikan S1 Ilmu Komputer di Universitas Udayana Bali dan S2 Ilmu Komputer di Universitas Gadjah Mada Yogyakarta. Dosen STMIK STIKOM Indonesia (STIKI Indonesia) di kota Denpasar. Mengajar mata kuliah Basis Data, Data Integration, Riset Teknologi Informasi, Tata Kelola Teknologi Informasi, e-Commerce.

BASIS DATA

Teori dan Perancangan

Sistem manajemen basis data telah bertransformasi dari sebelumnya hanya sebagai sistem penyimpanan data menjadi sistem yang memiliki kemampuan penyimpanan spesifik untuk permasalahan tertentu. Basis data telah digunakan secara luas baik pada aplikasi desktop, aplikasi berbasis website, bahkan aplikasi mobile. Ilmu yang mempelajari basis data telah menjadi ilmu dasar yang fundamental dalam pengembangan teknologi informasi.

Salah satu sistem basis data yang secara luas dimanfaatkan adalah basis data relasional. Basis data relasional banyak diterapkan dan digunakan dalam merancang sebuah perangkat lunak. Dengan pemahaman tentang data, informasi, atribut dan membangun relasi antar atribut serta perancangan basis data yang baik maka diharapkan dapat menghasilkan perangkat lunak yang baik sehingga dapat mempermudah proses penyimpanan data serta pencarian data yang diperlukan oleh penggunanya.

Pemodelan rancangan basis data relasional berbasis diagram telah lama diperkenalkan dan dikembangkan secara luas. Pengembangan pemodelan ini mengakibatkan ada beragam pemodelan yang diperkenalkan, akan tetapi secara konsepsi, model-model yang ada memiliki tujuan yang sama dan aturan yang hampir sama.

Tersedianya beragam model rancangan basis relasional data juga mengakibatkan adanya perbedaan dukungan model pada aplikasi perancangan basis data. Bahkan ada juga aplikasi yang mengembangkan model spesifik yang hanya tersedia pada aplikasi tersebut. Pemahaman tentang model diagram sangat diperlukan agar aplikasi dapat digunakan secara optimal. Sehingga perlu pemahaman yang mendalam tentang konsep dasar basis data berbasis pendekatan ER.



YAYASAN KITA MENULIS
press@kitamenulis.id
www.kitamenulis.id

ISBN 978-623-6761-31-1



9 786236 761311