Konsep dan Implementasi Basis Data

Zulfikar Yusya Mubarak, M.Kom.
Annisa Rahayu Pangesti, M.Kom.

UNAIC PRESS
CILACAP

KONSEP DAN IMPLEMENTASI BASIS DATA

UU No 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 Pelindungan Pasal 26

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

i. Penggunaan kutipan singkat Ciptaan dan/atau produk Hak Terkait untuk pelaporan peristiwa aktual yang ditujukan hanya untuk keperluan penyediaan informasi aktual;
ii. Penggandaan Ciptaan dan/atau produk Hak Terkait hanya untuk kepentingan penelitian ilmu pengetahuan;
iii. Penggandaan Ciptaan dan/atau produk Hak Terkait hanya untuk keperluan pengajaran, kecuali pertunjukan dan Fonogram yang telah dilakukan Pengumuman sebagai bahan ajar; dan
iv. 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 melakukan pelanggaran hak ekonomi sebagaimana dimaksud dalam Pasal 9 ayat (1) huruf i untuk Penggunaan Secara Komersial dipidana dengan pidana penjara paling lama 1 (satu) tahun dan/atau pidana denda paling banyak Rp100.000.000 (seratus 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 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).

KONSEP DAN IMPLEMENTASI BASIS DATA

Zulfikar Yusya Mubarak, M. Kom.

Annisa Rahayu Pangesti, M. Kom.

Konsep dan Implementasi Basis Data

Zulfikar Yusya Mubarak, Annisa Rahayu Pangesti

Desain Cover :
Rulie Gunadi

Sumber :
shutterstock (PeachShutterStock)

Tata Letak :
Ajuk

Proofreader :
Hanun Dinah Syafitri

Ukuran :
x, 82 hlm, Uk: 15.5x23 cm

ISBN :
978-623-88026-1-6

Cetakan Pertama :
Mei 2023

Hak Cipta 2023, Pada Penulis

Isi diluar tanggung jawab percetakan

Copyright © 2023 by Unaic Press
All Right Reserved

Hak cipta dilindungi undang-undang
Dilarang keras menerjemahkan, memfotokopi, atau
memperbanyak sebagian atau seluruh isi buku ini
tanpa izin tertulis dari Penerbit.

Unaic Press
Anggota IKAPI (254/JTE/2022)
Jl. Cerme No.24, Wanasari, Kabupaten Cilacap, Jawa Tengah 53223
Telp/Faks: (0282) 532975
Website: www.universitasalirsyad.ac.id
E-mail: humas@universitasalirsyad.ac.id

KATA PENGANTAR

Assalamu'alaikum wa rahmatullahi wa barakatuh,

Puji dan syukur kita panjatkan kepada Allah Swt. yang telah memberi rahmat dan hidayah-Nya sehingga revisi penyusunan buku ajar *Konsep dan Implementasi Basis Data* ini akhirnya bisa diselesaikan. Buku ajar ini disusun untuk menunjang perkuliahan dan sebagai pegangan wajib dalam mata kuliah Basis Data di Program Studi Sarjana Bisnis Digital Universitas Al-Irsyad Cilacap. Mata kuliah Basis Data merupakan salah satu mata kuliah yang diberikan kepada mahasiswa di semester 2. Mata kuliah ini mempelajari tentang konsep sistem Basis Data, jenis-jenis pemodelan Basis Data, bagaimana merancang Basis Data yang baik dan perkembangan Basis Data saat ini. Dalam kuliah diberikan contoh-contoh nyata perancangan dan implementasi Basis Data menggunakan SQL dan mengembangkan aplikasi menggunakan Basis Data.

Materi yang disajikan disesuaikan dengan rencana pembelajaran semester dari mata kuliah tersebut yang mencakup pengenalan Basis Data pada bab 1, pemodelan data pada bab 2, struktur dasar SQL pada bab 3 dan keamanan dan integritas data dari sisi islami pada bab 4.

Penulis menyadari masih banyak ketidaksempurnaan pada penulisan buku ajar ini, baik isi maupun redaksinya, oleh karenanya kritik dan saran yang membangun diharapkan dapat memperbaiki buku ajar ini untuk tahun-tahun berikutnya.

Terima kasih kepada semua pihak yang telah membantu baik secara langsung ataupun tidak terhadap terselesaikannya buku ajar ini. Akhir kata, buku ajar ini dapat bermanfaat bagi siapa saja yang membutuhkannya.

Wassalamu'alaikum wa rahmatullahi wa barakatuh,

Penulis.

DAFTAR ISI

BAB 1
PENGENALAN BASIS DATA

Pokok Bahasan
- Konsep dan definisi Basis Data
- Kelebihan Basis Data
- Kriteria Basis Data
- Penerapan Basis Data
- Dalam Program/Aplikasi

1.1. PENDAHULUAN

Pada era digital sekarang ini, Data menjadi hal yang penting dan mempunyai dampak yang besar terhadap kehidupan manusia, terlebih dalam hal perkembangan teknologi. Segala aktivitas di hampir semua bidang tidak terlepas dari data, seperti bidang pendidikan, kesehatan, hiburan maupun bisnis.

Pada bidang pendidikan misalnya, mahasiswa menggunakan Basis Data yang tersimpan dalam Sistem Akademik milik kampus mereka untuk melihat jadwal perkuliahan, nilai ujian, tugas, materi-materi pembelajaran yang dibagikan oleh dosen dan aktivitas lainnya yang bersumber dari data.

Begitu pula dengan aktivitas yang terjadi dalam bidang bisnis atau perkantoran. Pengembangan Sistem Informasi di suatu bisnis atau perusahaan memerlukan Basis Data sebagai penyimpan data. Karyawan melakukan pengolahan data yang tercatat di lembar kertas dan juga menggunakan sistem informasi. Pemanfaatan Basis Data dapat meningkatkan kinerja perusahaan & meningkatkan daya saing perusahaan tersebut.

Pada Bab 1 ini, materi yang dibahas dimulai dari mengenal konsep apa itu Basis Data dengan mempelajari definisinya, lalu pembahasan tentang kelebihan, kriteria, sistem dan arsitektur Basis Data, serta penerapan Basis Data dalam Program/Aplikasi yang sering digunakan di era digital ini.

1.2. TUJUAN PEMBELAJARAN

Setelah mempelajari bab ini, Anda diharapkan mampu:

a. CPMK

Menjelaskan konsep Basis Data dalam pembuatan sebuah program

b. Sub CPMK

1. Menjelaskan konsep dan definisi Basis Data.
2. Menjelaskan kelebihan dan kriteria Basis Data.
3. Menganalisis penerapan Basis Data dalam suatu program/aplikasi.

1.3. METODE PERKULIAHAN

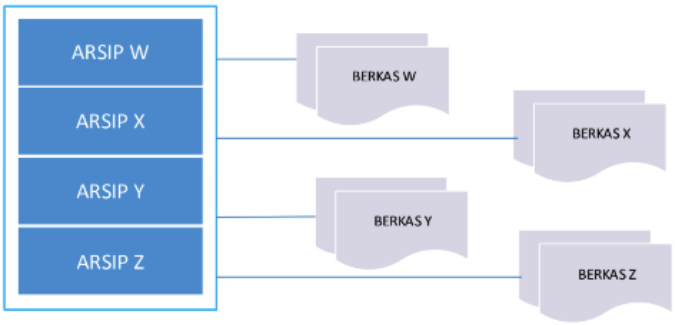
Pada bab 1 ini, metode pembelajaran yang digunakan adalah *Active Learning* dan *Small Group Dissusion*.

1.4. MATERI POKOK BAHASAN

1.4.1. KONSEP DAN DEFINISI BASIS DATA

Suatu Basis Data merupakan kumpulan data terpusat dan terstruktur yang disimpan dalam sistem komputer, di mana sistem tersebut menyediakan fasilitas untuk mengambil, menambahkan, memodifikasi dan menghapus data saat diperlukan. Sistem juga menyediakan fasilitas untuk dapat mengubah data yang diambil menjadi informasi yang berguna.

Basis Data (*database*) dapat dibayangkan sebagai sebuah lemari arsip yang terdiri dari beberapa rak yang digunakan untuk menyimpan arsip, Konsep Basis Data dapat dianalogikan sebagai berikut: Basis Data = Lemari Arsip = Lemari Pakaian = Rak Buku = Rak Piring = dan lain sebagainya yang pasti dikelola untuk menyimpan barang-barang tertentu. Seperti pada gambar 1.1.



Gambar 1.1. Lemari Arsip

Jika kita memiliki sebuah lemari arsip dan bertugas untuk mengelolanya, maka beberapa hal yang kemungkinan besar akan kita lalukan untuk mengelola tempat penyimpanan tersebut, yaitu:

a. Memberi sampul/map pada kumpulan arsip yang disimpan
b. Menentukan kelompok/jenis arsip
c. Memberi penomoran dengan pola tertentu yang nilainya unik pada setiap sampul/map maupun penomoran kelompok atau jenis arsip
d. Menempatkan arsip tersebut dengan cara/urutan tertentu di lemari arsip

Tujuannya adalah jika pada saat data atau arsip tersebut dibutuhkan, maka dapat ditemukan dan diperoleh dengan cepat dan mudah.

Pengelolaan data dalam bentuk seperti ini tidak akan menjadi masalah ketika berkas masih dalam jumlah yang sedikit, namun ketika sudah semakin banyak data yang disimpan dalam waktu yang lama, maka permasalahan akan muncul salah satunya di saat akan mencari kembali data yang tersimpan. Oleh karena itu dibutuhkan suatu sistem yang dapat menyimpan, mengorganisir, dan mengambil data kembali saat data diperlukan yang dapat disebut dengan Basis Data.

Dalam Basis Data, data dan informasi merupakan bagian yang penting, sehingga data dan informasi memiliki pengertian: Data merupakan nilai (*value*) yang merepresentasikan deskripsi dari suatu objek atau kejadian (*event*). Sedangkan Informasi merupakan hasil dari pengolahan data dalam suatu bentuk yang lebih berguna dan lebih berarti bagi penerimanya serta menggambarkan suatu kejadian-kejadian yang nyata (*fact*) yang digunakan untuk pengambilan keputusan. Data lebih bersifat historis, sedangkan informasi mempunyai tingkatan yang lebih tinggi, lebih dinamis, serta mempunyai nilai sangat penting.

Basis Data adalah suatu kumpulan data terhubung yang disimpan secara bersama-sama pada suatu media, yang diorganisasikan berdasarkan sebuah skema atau struktur tertentu, dan dengan software untuk melakukan manipulasi untuk kegunaan tertentu (Nugroho, 2004).

Operasi dasar Basis Data:

a. *Create database*
b. Drop database
c. Create table
d. Drop table
e. Insert
f. Retrieve/Search
g. Update
h. Delete

Sedangkan menurut Fathansyah, Basis Data adalah:

"*Himpunan kelompok data* (*arsip*) *yang saling berhubungan yang diorganisasi sedemikian rupa agar dapat dimanfaatkan kembali dengan cepat dan mudah*." (Fathansyah, 2015)

Tidak semua bentuk penyimpanan data secara elektronis bisa disebut Basis Data. Tidak sekadar penyimpanan secara elektronis.

Pada penyimpanan dokumen berisi data dalam *file* teks (dengan program pengolahan kata), *file spreadsheet*, dll., tidak ada pemilihan dan pengelompokan data sesuai jenis/fungsi data, sehingga akan menyulitkan pencarian data kelak. Keutamaan Basis Data adalah pengaturan/ pemilihan/pengelompokan/pengorganisasian data yang akan disimpan sesuai fungsi dan jenisnya.

1.4.2. KELEBIHAN BASIS DATA

Pemanfaatan Basis Data dapat memberikan beberapa kelebihan yaitu:

1. Kecepatan dan kemudahan (*speed*)

Pemanfaatan Basis Data memungkinkan kita menyimpan, memanipulasi, menampilkan data dengan lebih cepat dan mudah.

2. Efisiensi ruang penyimpanan (*space*)
   Penekanan redudansi data dengan menerapkan sejumlah teknik pengkodean atau membuat relasi-relasi antar *file*, akan mengurangi pemakaian *space* di media penyimpanan.
3. Keakuratan (*accuracy*)
   Keakuratan dengan penerapan aturan/batasan (*constraint*) tipe data, domain data, keunikan data, dsb dalam pemasukan/penyimpanan data contoh: umur, jenis kelamin, email, telepon dan nilai.
4. Ketersediaan (*availability*)
   Bersifat tersedia ketika akan digunakan.
5. Kelengkapan (*completeness*)
   Kelengkapan dengan menyimpan struktur yang mendefinisikan objek-objek dalam Basis Data maupun definisi detail dari tiap objek.
6. Keamanan (*security*)
   Penerapan keamanan Basis Data berkaitan dengan pemberian hak akses kepada *user* dengan tingkatan tertentu beserta operasi–operasi yang boleh dilakukan.
7. Kebersamaan pemakaian (*sharability*)
   Kebersamaan pemakaian data umumnya ada dalam Basis Data yang menggunakan aplikasi multiuser.

1.4.3. KRITERIA BASIS DATA

Basis Data mempunyai beberapa kriteria penting yang harus dipenuhi, yaitu:

1. Bersifat *data oriented* dan bukan *program oriented* yang akan menggunakannya
2. Data dapat digunakan oleh pemakai yang berbeda-beda atau beberapa program aplikasi tanpa perlu mengubah Basis Datanya (*multiple user*)
3. Data dalam Basis Data dapat berkembang dengan mudah, baik volume maupun strukturnya
4. Data yang ada dapat memenuhi kebutuhan sistem-sistem baru secara mudah
5. Data dapat digunakan dengan cara yang berbeda
6. Kerangkapan data (*data redundancy*) minimal

1.4.4. PENERAPAN BASIS DATA DALAM PROGRAM/APLIKASI

Bidang fungsional banyak memanfaatkan Basis Data demi keefisiensian, akurasi dan kecepatan operasi dalam menjalankan prosedur, contoh pengelolaan Basis Data dalam bidang fungsional yaitu:

1. Kepegawaian (dengan banyak pegawai)
2. Pergudangan/*inventory* (pabrik, grosir, apotek dan lain-lain)
3. Akuntasi (perusahaan)
4. Reservasi (hotel, pesawat, kereta api dan lain-lain)
5. Layanan pelanggan (perusahaan dengan banyak pelanggan)

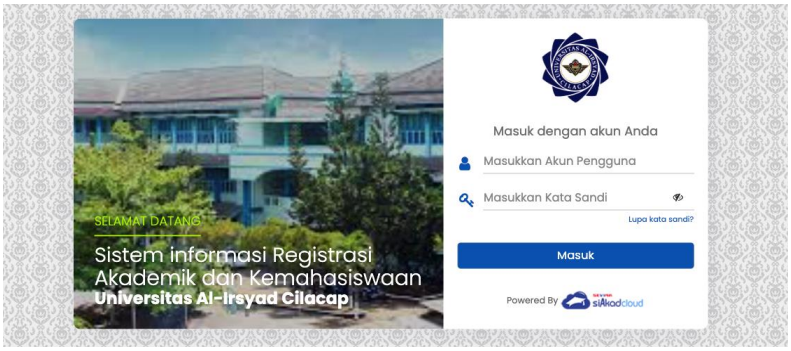
Bentuk organisasi/perusahaan yang memanfaatkan Basis Data (sebagai komponen sistem informasi organisasi/perusahaan) dapat berupa:

1. Perbankan (pengelolaan data nasabah, data tabungan, data pinjaman, pembuatan laporan akuntansi, dan lain-lain)
2. Asuransi (pengelolaan data nasabah, data pembayaran premi, pemrosesan pengajuan klaim dan lain-lain)
3. Rumah sakit (pengelolaan data pasien, histori penyakit, data dokter, data jadwal praktik dokter, data pembayaran perawatan dan lain-lain)Produsen Barang (pengelolaan data keluar masuk barang, stok barang, dan lain-lain)
4. Industri manufaktur (pengelolaan pesanan barang, data karyawan, dan lain-lain)
5. Pendidikan/sekolah (data siswa, jadwal kuliah, nilai, dan lain-lain)
6. Telekomunikasi (data administrasi, data pelanggan, dan lain-lain)

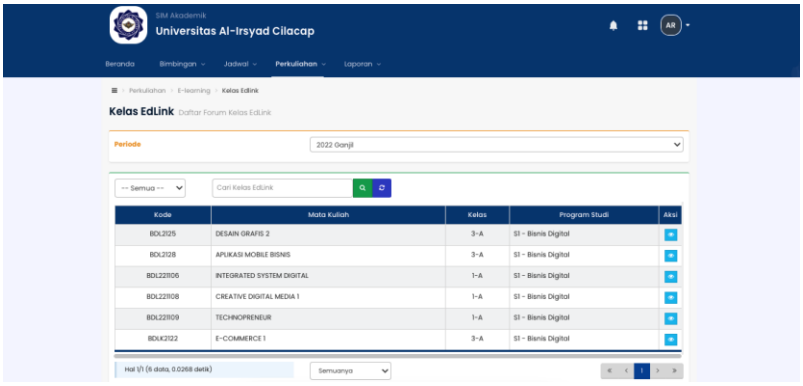
Salah satu contoh nyata penerapan pengolahan Basis Data yang bisa dialami secara langsung adalah pada Sistem Informasi Registrasi Akademik dan Kemahasiswaan (SIRAKA) Universitas Al-Irsyad Cilacap.

SIRAKA merupakan salah satu contoh sistem informasi yang memanfaatkan Basis Data dalam penyampaian informasinya. Banyak *user* yang dapat memanfaatkan informasi yang ada di sistem, mahasiswa, dosen, kepala tata usaha dan lain-lain. Gambar 1.2. menunjukkan halaman *website* dengan *login* sebagai dosen.

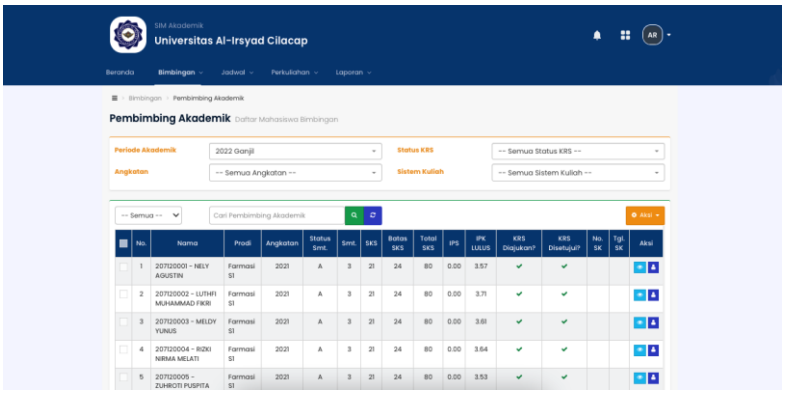
Beberapa informasi yang diberikan untuk dosen meliputi info mata kuliah yang diampu, daftar mahasiswa bimbingan akademik, pengelolaan nilai dsb. Informasi ini tentu didapat dari sekumpulan data yang disimpan. Sedangkan gambar 1.3 menunjukkan halaman mata kuliah yang diampu.



Gambar 1.2. Sistem Informasi Registrasi Akademik dan Kemahasiswaan UNAIC



Gambar 1.3. Halaman Kelas EdLink Mata Kuliah Diampu-Siraka



Gambar 1.4. Halaman Pembimbing Akademik-SIRAKA

Gambar 1.4 menunjukkan daftar mahasiswa bimbingan akademik, berupa NIM, nama mahasiswa, program studi dan juga termasuk dosen bimbingan akademik dapat melihat transkrip nilai dari tiap mahasiswa bimbingan akademiknya.

1.5. RANGKUMAN

Basis Data adalah suatu susunan/kumpulan data operasional lengkap dari suatu organisasi/perusahaan yang diorganisir/dikelola dan simpan secara terintegrasi dengan menggunakan metode tertentu dengan menggunakan komputer sehingga mampu menyediakan informasi yang diperlukan pemakainya. Basis Data memiliki beberapa kelebihan yang dapat membantu pengguna yaitu dalam segi: kecepatan dan kemudahan, (*speed*), efisiensi ruang penyimpanan (*space*), keakuratan (*accuracy*), ketersediaan (*availability*), kelengkapan (*completeness*), keamanan (*security*), kebersamaan pemakaian (*sharability*).

1.6. LATIHAN SOAL

Sebagai Latihan untuk mencapai kompetensi yang diharapkan, silakan kerjakan beberapa soal di bawah ini dengan jawaban yang tepat:

1. Berikan beberapa contoh penerapan Basis Data di bidang usaha bisnis!
2. Carilah contoh-contoh program/aplikasi yang sudah ada dan banyak digunakan dari penerapan Basis Data tersebut lalu lengkapi dengan *screenshoot*, foto, atau tampilan-tampilan yang mendukung!
3. Analisislah data dan informasi apa saja yang Anda anggap dihasilkan dari program/aplikasi yang berlandaskan Basis Data tersebut!

1.7. KISI-KISI JAWABAN

1. *Database* bagi perusahaan memiliki peran sangat signifikan, baik itu untuk pengambilan keputusan dengan bantuan DSS (*Decision Support System*) yang sudah terbangun, untuk memberikan *Value Added* bagi *customer* dengan kemampuannya memberikan informasi yang akurat tepat dan *up to date*, dan lain sebagainya.

2. Siberuang-sistem ruang

3. Penggunaan teknologi *database* di dunia bisnis bermanfaat menghemat waktu dan biaya karena dengan *database* yang terkomputerisasi kita bisa banyak menyimpan informasi seperti mencetak, memuat, menampilkan data yang akurat, memudahkan pengaksesan data, mengisolasi data untuk di standardisasikan, mengurangi redundasi data dan inkonsistensi. Dan yang menjadi faktor pertimbangan bagi para pelaku bisnis dalam skala besar adalah apabila desain yang dibangun tidak cermat dapat menyebabkan hilangnya data yang di butuhkan, data yang tidak konsisten, proses *update* yang lambat dan lain-lain.

BAB 2
SISTEM BASIS DATA

Pokok Bahasan
- Definisi Sistem
- Sistem Basis Data (SBD)
- Komponen Sistem Basis Data
- *Database Management System* (DBMS)
- Arsitektur Sistem Basis Data

2.1. PENDAHULUAN

Dalam proses pengolahan Basis Data (*database*) yang datanya semakin banyak dan beragam, perangkat lunak (*software*) sangat dibutuhkan untuk dapat membantu menyimpan, mengorganisir dan mengambil kembali data dari *database. Software* ini sering disebut dengan Sistem Manajemen Basis Data (*Database Management System/*DBMS). Sistem Basis Data memiliki lingkup yang lebih luas dibanding Basis Data itu sendiri.

Pada Bab 2 ini, materi yang dibahas yaitu mencakup definisi sistem, sistem Basis Data, komponen sistem Basis Data, sistem manajemen Basis Data, dan arsitektur Basis Data.

2.2. TUJUAN PEMBELAJARAN

Setelah mempelajari bab ini, Anda diharapkan mampu:

a. CPMK

Menjelaskan konsep sistem Basis Data dalam pengembangan suatu program

b. Sub CPMK

1. Menjelaskan konsep dan definisi sistem Basis Data.
2. Menjelaskan komponen sistem Basis Data
3. Menjelaskan sistem manajemen Basis Data (*database management system*) dan arsitektur sistem Basis Data.

2.3. MATERI POKOK BAHASAN

2.3.1. DEFINISI SISTEM

Sistem, menurut Fathansyah dalam bukunya Basis Data (Revisi Ketiga) didefinisikan sebagai *"sebuah tatanan yang terdiri atas sejumlah komponen fungsional* (*dengan satuan fungsi dan tugas khusus*) *yang saling berhubungan & secara bersama-sama bertujuan untuk memenuhi suatu proses tertentu."* (Fathansyah, 2015)

Contoh sistem yang sering kita lihat sehari-hari seperti Sistem kendaraan, yang mempunyai komponen-komponen seperti: komponen starter, komponen perapian, komponen penggerak, pengerem, pelistrikan, spedometer, lampu, dll. Komponen-komponen ini saling terhubung dan mempunyai tujuan yang sama yaitu untuk membuat kendaraan tersebut bisa dikendarai dengan nyaman & aman.

2.3.2. SISTEM BASIS DATA

Sistem Basis Data memiliki beberapa definisi antara lain:

a. Sistem Basis Data merupakan sekumpulan Basis Data dengan para pemakai yang menggunakan Basis Data secara bersama-sama, personil yang merancang dan mengelola Basis Data, teknik-teknik untuk merancang dan mengelola Basis Data, serta sistem komputer yang mendukungnya.

b. Sistem yang terdiri atas kumpulan *file*/tabel yang saling berhubungan (dalam sebuah Basis Data di sebuah sistem komputer) dan sekumpulan program (DBMS) yang memungkinkan beberapa pemakai dan/atau program lain untuk mengakses dan memanipulasi *file-file* (tabel-tabel) tersebut.

2.3.3. KOMPONEN SISTEM BASIS DATA

Sistem Basis Data memiliki komponen-komponen penting yaitu:

1. Perangkat keras

   Perangkat keras yang biasanya terdapat dalam sebuah Sistem Basis Data:

   a. Komputer: 1 untuk sistem yang *stand-alone* dan untuk yang lebih dari 1 sistem jaringan
   b. Memori sekunder yang *on-line* (*Harddisk*)
   c. Memori sekunder yang *off-line* (*Removable Disk*)
   d. keperluan *backup* data
   e. Media/perangkat komunikasi (untuk sistem jaringan)

2. Sistem operasi

   Secara sederhana merupakan program yang mengaktifkan sistem komputer, mengendalikan seluruh sumber daya (*resource*) dalam komputer dan melakukan operasi-operasi dasar dalam komputer. Program DBMS/SMBD hanya dapat aktif (*running*) jika sistem operasi yang dikehendakinya (sesuai) telah aktif. Contoh: Untuk komputer *stand-alone* dan *client:* Windows, Linux untuk komputer server: Windows Server, Unix, Linux

3. Basis Data

   1 Sistem Basis Data dapat memiliki beberapa Basis Data. Setiap Basis Data dapat berisi sejumlah objek

seperti: Tabel, Indeks, dll. Setiap Basis Data juga menyimpan definisi struktur (baik untuk BD maupun objek-objeknya secara rinci)

4. Sistem pengelola Basis Data (DBMS)
   DBMS akan menentukan bagaimana data diorganisasi, disimpan, diubah dan diambil kembali. DBMS juga menerapkan mekanisme pengamanan data, pemakaian data secara bersama, pemaksaan keakuratan/konsistensi data, dsb. Contoh: dBase, MS-Access, MS-SQL Server, Oracle Database, MySQL.
5. Pemakai (Programmer, User mahir, user umum, user khusus)
   Berdasarkan cara berinteraksi, pemakai SBD dapat dibedakan menjadi:

   Programmer Aplikasi

   Pemakai yang berinteraksi melalui *Data Manipulation Language* (DML), yang disertakan (*embedded*) dalam program yang ditulis dalam bahasa pemrograman induk (seperti C, C++, PHP, Pascal, Java, dll)

   ***User* Mahir (*Casual User*)**

   Pemakai yang berinteraksi tanpa menulis modul program. Mereka menyatakan *query* dengan bahasa yang telah disediakan oleh DBMS.

   ***User* Umum (*End User/Naive User*)**

   Pemakai yang berinteraksi melalui pemanggilan satu program aplikasi permanen (*executable* program) yang telah disediakan sebelumnya. Contoh:

   Seorang mahasiswa memasukkan data KRS ke dalam BD dengan menggunakan aplikasi KRS online.

   ***User Khusus* (*Specialized User*)**

   Pemakai yang menulis aplikasi BD nonkonvensional, tetapi untuk keperluan khusus, seperti untuk aplikasi

Artificial Intelligence, Sistem Pakar, Pengolahan Citra, dll.

2.3.4. DATABASE MANAGEMENT SYSTEM (*DBMS*)

Program komputer yang digunakan untuk memasukkan, mengubah, menghapus, memanipulasi, dan memperoleh data informasi dengan praktis dan efisien termasuk juga mengatur mekanisme pengamanan data, pemakaian data bersama, pemaksaan keakuratan/konsistensi data dan sebagainya. Berikut ini merupakan kelemahan dan kelebihan DBMS.

Keuntungan DBMS, sebagai berikut:

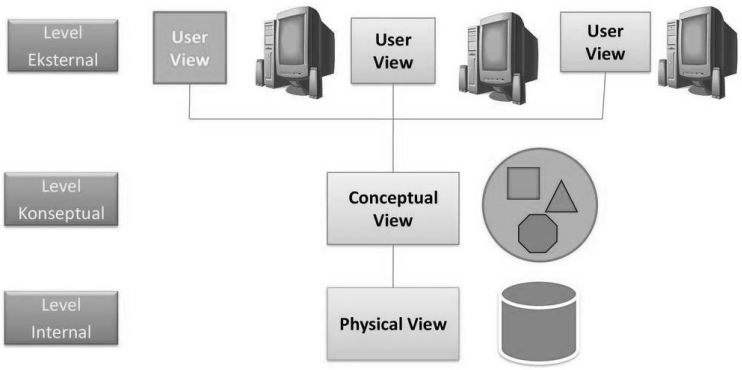
1. Pengontrolan kerangkapan data-konsistensi data
2. Lebih banyak informasi dari jumlah data yang sama-*sharing* data
3. Peningkatan integrasi data-peningkatan keamanan
4. Penegakan standar layanan

Kekurangan DBMS, sebagai berikut:

1. Kompleksitas-Ukuran
2. Biaya DBMS-biaya perangkat keras tambahan
3. Biaya konversi teknologi-performa
4. Dampak kegagalan yang lebih besar

2.3.5. ARSITEKTUR BASIS DATA

Salah satu tujuan DBMS: menyediakan antarmuka (*interface*) yang ramah (*user-friendly*) Maka, DBMS akan menyembunyikan detail tentang bagaimana data disimpan dan dikelola. Sehingga, data yang terlihat oleh pemakai dapat berbeda dengan yang sesungguhnya tersimpan secara fisik. Abstraksi Data mengacu pada tingkatan/level dalam bagaimana melihat data dalam sebuah Sistem Basis Data.



Gambar 2.1. Level Abstraksi Data

3. Level Fisik (*Physical Level*)
   Level terendah dalam abstraksi data yang menunjukkan bagaimana sesungguhnya suatu data disimpan. Pemakai berkompeten mengetahui bagaimana representasi fisik dari penyimpanan/pengorganisasian data.
4. Level *Logical* (*Conceptual Level*)
   Menjabarkan data apa saja yang sesungguhnya disimpan dalam *database* dan mendeskripsikan hubungan antar data. Menggambarkan data apa yang sebenarnya secara fungsional disimpan dalam Basis Data dan hubungannya dengan data lain.
5. Level Penampakan (View Level)
   Data yang dibutuhkan oleh setiap *user* dapat berbeda-beda & kemungkinan hanya mencakup sebagian data dalam Basis Data yang kemunculannya diatur oleh aplikasi *end user*. Data yang diperoleh bahkan dapat sama sekali berbeda dengan representasi fisiknya supaya dapat bermakna bagi *user*.

2.4. RANGKUMAN

Sistem Basis Data terdiri dari kumpulan *file* atau tabel dan saling berhubungan dan memungkinkan beberapa *user* untuk mengakses dan memanipulasi *file* atau data dan memiliki lingkup yang lebih luas dibanding Basis Data itu sendiri. Terdapat 5 komponen Sistem Basis Data yaitu: Perangkat Keras (*Hardware*), Sistem Operasi, Basis Data, Sistem Manajemen Basis Data (SMBD)/*Database Management System* (DBMS) dan Pemakai (*User*). Dengan adanya DBSM pengguna bisa membuat, mengontrol, memelihara, dan mengakses Basis Data secara mudah dan efisien. Karena pada umumnya DBMS memiliki fitur-fitur yang memungkinkan data dapat diakses dengan mudah.

2.5. LATIHAN SOAL

Sebagai Latihan untuk mencapai kompetensi yang diharapkan, silakan kerjakan beberapa soal di bawah ini dengan jawaban yang tepat:

1. Berikan beberapa contoh Sistem Basis Data!
2. Analisislah data apa saja yang termasuk dalam level fisik, *logical*, dan level penampakan dari sistem Basis Data tersebut!

2.6. KISI KISI JAWABAN

1. Microsoft SQL Server, Oracle **Database**, MySQL, PostgreSQL dan IBM Db2.

2. **Level Fisik**

Level terendah dalam abstraksi data, yang menunjukkan bagaimana sesungguhnya suatu data disimpan

Level *Logical*

Level terendah dalam abstraksi data, yang menunjukkan bagaimana sesungguhnya suatu data disimpan

Level Penampakan

Pandangan para pengguna *database* pada masing-masing pengguna *database*, sehingga memiliki cara pandang yang berbeda-beda tergantung pada macam data apa saja yang tersedia atau dapat diakses oleh pengguna.

BAB 3
PEMODELAN DATA

Pokok Bahasan
- Definisi Pemodelan Data
- Tujuan Pemodelan Data
- Jenis Model Data
- *Entity Relationship Diagram* (ERD)
- *Unified Modelling Language* (UML)
- Agregasi, Struktur Tabel, Normalisasi
- *Data Definition Language* (DDL)

3.1. PENDAHULUAN

Pemodelan adalah proses untuk membuat sebuah model dari sistem. Model adalah representasi dari sebuah bentuk nyata, sedangkan sistem adalah saling keterhubungan antar elemen yang membangun sebuah kesatuan, biasanya dibangun untuk mencapai tujuan tertentu. Tujuan suatu pemodelan adalah untuk menganalisis dan memberi prediksi yang dapat mendekati kenyataan sebelum sistem di terapkan di lapangan. Kesulitan untuk memprediksi dan mengamati proses tertentu pada lapangan akan menjadi persoalannya, di mana model dapat memformulasikan sebuah proses tertentu namun tidak memungkinkan untuk melakukan analisis untuk mendapatkan solusi tepat sehingga perlu dilakukan lagi perbandingan atau validasi antara pemodelan matematik dengan kondisi lapangan.

3.2. TUJUAN PEMBELAJARAN

Setelah mempelajari bab ini, Anda diharapkan mampu:

a. CPMK

Melakukan pemodelan data dan mendemonstrasikan penggunaan SQL secara interaktif untuk melakukan berbagai operasi terhadap Basis Data

b. SUB CPMK

1. Menjelaskan definisi dan tujuan pemodelan data.
2. Menjelaskan jenis-jenis model data.
3. Menjelaskan tentang ERD, UML, Agregasi, Struktur Tabel, Normalisasi, dan DDL.

3.3. MATERI POKOK BAHASAN

3.3.1. DEFINISI PEMODELAN DATA

Model data adalah suatu konsep yang terintegrasi dalam menggambarkan hubungan (*relationships*) antar data dan batasan-batasan (*constraint*) data dalam suatu sistem *database.*

3.3.2. TUJUAN PEMODELAN DATA

Pemodelan data memberi Anda kesempatan untuk memahami data serta memilih teknologi yang tepat untuk menyimpan dan mengelola data ini. Dengan cara yang sama seperti saat arsitek mendesain cetak biru sebelum membangun sebuah rumah, pemangku kepentingan bisnis mendesain model data sebelum mereka merekayasa solusi Basis Data untuk organisasi mereka.

Pemodelan data memberikan manfaat berikut:

1. Mengurangi kesalahan dalam pengembangan perangkat lunak Basis Data
2. Memfasilitasi kecepatan dan efisiensi desain serta pembuatan Basis Data
3. Menciptakan konsistensi dalam dokumentasi data dan desain sistem di seluruh organisasi

4. Memfasilitasi komunikasi antara perekayasa data dan tim kecerdasan bisnis

3.3.3. JENIS MODEL DATA

1. Model data berbasis objek

   Model data berbasis objek menggunakan konsep entitas, atribut dan hubungan antar entitas. Model data berbasis objek terdiri dari: *ENTITY RELATIONSHIP MODEL, BINARY MODEL, SEMANTIK DATA MODEL* dan *INFOLOGICAL MODEL.* Namun di sini yang akan sedikit dibahas hanyalah *ENTITY RELATIONSHIP MODEL SEMANTIC dan SEMANTIK DATA MODEL.*

   a. *ENTITY RELATIONSHIP MODEL*

      Digunakan untuk menjelaskan hubungan antar data dalam *database* atas dasar anggapan bahwa *real word* terdiri dari *object-object* dasar di mana *object-object* tersebut memiliki relasi atau keterhubungan. Dalam ER MODEL terdapat istilah *MAPPING CARDINALITY* yaitu jumlah *entity* yang dapat dikaitkan dengan *entity* lainnya melalui *relation self.*

   b. *SEMANTIC MODEL*

      Hampir sama dengan ER MODEL, perbedaannya hanya terletak pada pernyataan adanya relasi antar objeknya. Jika pada ER MODEL menyatakan adanya relasi antar objek menggunakan symbol-simbol namun pada *SEMANTIC MODEL* menggunakan kata-kata.

2. Model data berbasis *record*

Digunakan untuk menguraikan seluruh logika dalam struktur *database*, juga digunakan untuk menguraikan implementasi dari *system database.* Hal itulah yang membedakan Model data berbasis *record* dengan model data berbasis objek. Dalam model data berbasis *record* kita mengenal 3 jenis data model, yaitu:

a. *Relational model*

b. Hierarki model

c. *Networking model*

3. Model data fisik

   Konsep bagaimana data disimpan pada media penyimpanan (storage) dalam suatu susunan secara fisik.

4. Model data konseptual

   Model konseptual bukanlah pendekatan proses informasi seorang *programmer* aplikasi, tetapi merupakan kombinasi beberapa cara untuk memproses data untuk beberapa aplikasi. Model konseptual tidak tergantung pada aplikasi individual, tidak tergantung pada DBMS yang digunakan, tidak tergantung pada *hardware* yang digunakan serta tidak tergantung juga pada *phisical model*. Pendekatan yang dilakukan pada perancangan model konseptual adalah dengan menggunakan model data *relational,* yaitu dengan Teknik normalisasi

3.3.4. *ENTITY RELATIONSHIP DIAGRAM* (ERD)

Entity Relationship Diagram (ERD) adalah suatu diagram yang digunakan untuk merancang suatu Basis Data, dipergunakan untuk memperlihatkan hubungan atau relasi antar entitas atau objek yang terlihat beserta atributnya. Bagi mahasiswa-mahasiswa yang masuk di jurusan komputer pasti tidak asing dengan istilah ERD, berikut ini contoh cara membuat ERD tahapan dan contoh studi kasus, tetapi sebelumnya kita pelajari lebih dalam tentang ERD. Di pembahasan sebelumnya sudah dijelaskan pengertian tentang model data. ERD dipelajari karena memang bertujuan untuk membantu para pengembang sistem dalam merancang relasi antar tabel dalam membuat *database,* jadi sebenarnya ERD itu adalah bakal calon menjadi tabel, kalau rancangan ERD benar maka rancangan *database* juga akan menjadi benar. Berbicara tentang *database* ada banyak sekali DBMS (*Database Management System*) baik yang *open source* maupun yang berbayar contohnya adalah MySQL, walaupun free tetapi memiliki kemampuan yang tidak kalah dengan DBMS yang berbayar dan MySQL memiliki berbagai macam *storage engine.* Untuk DBMS yang berbayar salah satu contohnya SQLServer dari Microsoft. Terlepas dari *database* bahwa objek utama dari pembuatan diagram ERD menunjukkan objek-objek (himpunan entitas) apa saja yang ingin dilibatkan dalam sebuah Basis Data dan bagaimana hubungan yang terjadi antara objek-objek tersebut. Berikut tahapan cara membuat ERD yang dijabarkan satu-persatu.

3.3.5. *UNIFIED MODELLING LANGUAGE* (UML)

UML adalah sekumpulan alat yang digunakan untuk melakukan abstraksi terhadap sebuah sistem atau perangkat lunak berbasis objek. UML merupakan singkatan dari *Unified Modeling Language*. UML juga menjadi salah satu cara untuk mempermudah pengembangan aplikasi yang berkelanjutan. Aplikasi atau sistem yang tidak terdokumentasi biasanya dapat menghambat pengembangan karena *developer* harus melakukan penelusuran dan mempelajari kode program. UML juga dapat menjadi alat bantu untuk *transfer* ilmu tentang sistem atau aplikasi yang akan dikembangkan dari satu *developer* ke *developer* lainnya. Tidak hanya antar *developer* terhadap orang bisnis dan siapa pun dapat memahami sebuah sistem dengan adanya UML.

Structural Diagram adalah:

a. *Class Diagram*, diagram ini terdiri dari *class*, *interface*, *association*, dan *collaboration*. Diagram ini menggambarkan objek-objek yang ada di sistem.
b. *Object Diagram*, diagram ini menggambarkan hasil instansi dari *class diagram*. Diagram ini digunakan untuk membuat *prototype*
c. *Component Diagram*, diagram ini menggambarkan kumpulan komponen dan hubungan antar komponen. Komponen terdiri dari *class*, *interface*, atau *collaboration*
d. *Deployment Diagram*, diagram ini menggambarkan kumpulan *node* dan hubungan antar *node*. *Node* adalah entitas fisik di mana

komponen di-*deploy*. Entitas fisik ini dapat berupa *server* atau perangkat keras lainnya.

3.3.6. *AGREGASI* (*AGGREGATION*)

Merupakan suatu proses untuk mendapatkan nilai dari kumpulan data yang sudah dikelompokkan. Pengelompokan data itu didasarkan pada kolom atau kombinasi kolom yang telah dipilih.

Berikut macam-macam Agregasi dalam SQL:

d. ***AVG* (*Average*)**

Fungsi AVG adalah suatu fungsi yang dapat digunakan untuk menghitung nilai rata-rata dari sebuah kolom.

Contoh: *select AVG* (*nama_field*) *from nama_table;*

Fungsi *COUNT* adalah fungsi yang digunakan dalam SQL untuk menghitung jumlah atau banyaknya baris data dari sebuah tabel.

Contoh: *select COUNT* (*) *from nama_table;*

Fungsi MAX adalah suatu fungsi yang digunakan dalam bahasa SQL untuk mencari sebuah nilai terbesar dari kumpulan data angka pada suatu kolom atau *field.*

Contoh: *select MAX* (*nama_field*) *from nama_table;*

Fungsi MIN adalah suatu fungsi yang digunakan dalam SQL untuk mencari nilai terkecil dari sebuah kumpulan data angka dalam suatu *field.*

Contoh: *select MIN* (*nama_field*) *from nama_table;*

Fungsi SUM adalah suatu fungsi yang digunakan dalam bahasa SQL untuk menghitung jumlah atau juga hasil penjumlahan data angka dari suatu *field* maupun kolom.

Contoh: *select SUM* (*nama_field*) *from nama _table;*

e. ***GROUP BY***

Sebuah perintah yang digunakan untuk mengelompokkan beberapa data dalam perintah *SELECT.*

Contoh: *SELECT * FROM* (*nama_tabel*) *GROUP BY atribut*

f. ***HAVING & WHERE***

Perintah SELECT yang digunakan untuk menentukan kondisi adalah *WHERE*, Namun untuk perintah *GROUP BY* ini yang digunakan adalah *HAVING*. *HAVING* juga biasanya dipakai untuk menyeleksi data berdasarkan fungsi *aggregate* tertentu (*min, max, avg, sum, count*). *WHERE* dipakai untuk menyeleksi data berdasarkan *comparison* (=,<>,<,>,<=,>=)

g. ***DISTINCT***

Distinct adalah sebuah perintah yang digunakan untuk *operator* di *database* MySQL. Bahkan hampir di semua *database* menggunakan ini untuk mencegah adanya duplikasi data atau *record.* Contoh: *SELECT DISTINCT nama_field FROM nama_tabel Fungsi nama_field ASC;*

h. ***CONCAT***

Concat Distinct adalah sebuah perintah yang digunakan untuk *menyambung* kata. Contoh:

select CONCAT (*nama_field," penghubung_string ", field_baru*) *from nama_table;*

3.3.7. *STRUKTUR* TABEL

Merupakan suatu tempat penyimpanan data. Penciptaan **tabel** dilakukan dengan menentukan **struktur tabel**. Field **struktur** disebut juga sebagai kolom atau atribut. Setelah **struktur** terbentuk selanjutnya dapat diisikan data pada setiap field.

3.3.8. *NORMALISASI*

Normalisasi data adalah elemen dasar data *mining* untuk memastikan *record* pada *dataset* tetap konsisten. Dalam proses normalisasi diperlukan transformasi data atau mengubah data asli menjadi format yang memungkinkan pemrosesan data yang efisien. Tujuan utama dari normalisasi data yakni menghilangkan redundansi data (pengulangan) dan menstandardisasi informasi untuk alur kerja data yang lebih baik. Normalisasi data digunakan untuk menskalakan data suatu atribut sehingga berada dalam rentang yang lebih kecil, seperti -1 hingga 1 atau 0 hingga 1. Hal ini umumnya berguna untuk algoritma klasifikasi. Teknik normalisasi data dalam data *mining* sangat membantu karena memberikan banyak manfaat, sebagai berikut:

a. Penerapan algoritma data *mining* menjadi lebih mudah
b. Algoritma data *mining* menjadi lebih efektif dan efisien
c. Data dapat diekstraksi dari *database* dengan lebih cepat

d. Data yang sudah di normalisasi memungkinkan untuk dianalisis dengan metode tertentu

3.3.9. *DATA DEFINITION LANGUAGE* (DDL)

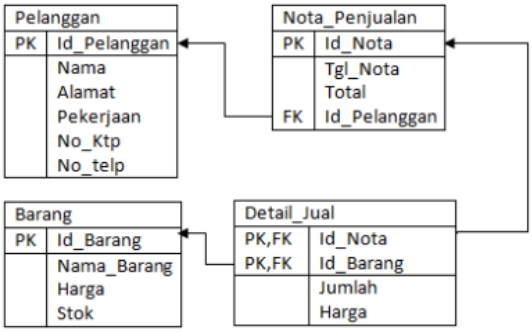
Data Definition Language (DDL) digunakan untuk mendefinisikan, mengubah dan menghapus Basis Data dan objek-objek yang diperlukan, misalnya tabel, *view, user, index* dan sebagainya. DDL biasa digunakan oleh DBA dalam pembuatan sebuah aplikasi Basis Data. Secara umum DDL yang digunakan ada empat, yaitu

a. *CREATE* untuk membuat objek baru.
b. *USE* untuk menggunakan objek.
c. *ALTER* untuk mengubah objek yang sudah ada.
d. *DROP* untuk menghapus objek.

3.4. LATIHAN SOAL

Sebagai Latihan untuk mencapai kompetensi yang diharapkan, silakan kerjakan beberapa soal di bawah ini dengan jawaban yang tepat:

1. Buatlah sebuah *database* baru dengan nama Penjualan_Barang
2. Gunakan *database* penjualan_barang kemudian buatlah tabel-tabel hasil dari diagram *relationship* penjualan barang.
3. Buatlah laporan praktikum dengan ketentuan sbb:
   a. Nama *file* laporan: PrakDB_Bab3_NIM.odt
   b. Isi *file* laporan: i. *Souce* sql ii. *Screenshot* CMD
   c. Simpan di direktori “PrakDB-NIM” ERD Penjualan Barang



ERD Penjualan Barang

BAB 4

KEAMANAN DAN INTEGRITAS DATA DARI SISI ISLAMI

Pokok Bahasan

- Hadis dan Ayat Al-Qur'an
- Keamanan Data
- Integritas Data
- Kebocoran Data

4.1. PENDAHULUAN

Selama ini sistem keamanan lebih banyak terfokus pada antivirus dan keamanan jaringan, namun kurang memperhatikan kerawanan dan ancaman yang berbahaya pada aplikasi dan *database* server. Sistem keamanan pada aplikasi dan *database server* merupakan kebutuhan yang tidak dapat dihindarkan lagi. Buku ini disusun untuk membantu pengguna teknologi informasi dalam menangani masalah keamanan aplikasi dan server *database*. Buku ini disusun secara interaktif dan tahap demi tahap, sehingga pembaca dengan bantuan CD-ROM yang interaktif sehingga bisa langsung dipraktikkan untuk membuat keamanan aplikasi dan server *database* yang telah disiapkan tanpa harus bertanya kepada instruktur.

4.2. TUJUAN PEMBELAJARAN

Setelah mempelajari bab ini, Anda diharapkan mampu:

1. CPMK

Memahami keamanan dan integritas data dari sisi islami

2. SUB CPMK

1. Menjelaskan hadis dan ayat Al-Qur'an yang membahas tentang keamanan dan integritas data
2. Menjelaskan tentang kebocoran data

4.3 MATERI POKOK BAHASAN

4.3.1 HADIST DAN AYAT AL-QUR'AN

Islam adalah agama yang banyak berbicara tentang keamanan. Beberapa di antaranya dapat kita temukan dalam Q.S. An-Nur ayat 27 yang mengatakan bahwa "Hai orang-orang yang beriman, janganlah memasuki rumah yang bukan rumahmu sehingga kamu minta izin dan memberi salam kepada penghuninya. Yang demikian itu lebih baik bagimu, agar kamu (selalu) ingat." Lanjut pada ayat 28 yang mengatakan "Dan jika kamu tidak menemui seorang pun di dalamnya, maka janganlah kamu masuk sebelum kamu mendapat izin. Dan jika dikatakan kepadamu, "kembalilah!" maka (hendaklah) kamu kembali. Itu lebih suci bagimu, dan Allah Maha Mengetahui apa yang kamu kerjakan."

Selanjutnya, dalam riwayat hadis yang disampaikan oleh Abu Hurairah bahwa Nabi Muhammad saw. bersabda "Apabila seseorang menengok atau melihat ke dalam rumahmu tanpa izin darimu, lalu kamu melemparnya dengan batu kerikil hingga tercungkil matanya, maka tidak ada dosa bagi kamu" (HR Al Bukhari dan Muslim). Selain itu, dalam QS Al-Kahfi ayat 90-98 terdapat dalam satu

penggalan kisah yang menceritakan konsep keamanan pada masa Nabi Zulkarnaen a.s. dengan bangsa Ya'juj dan Ma'juj. Pada masa itu, Nabi Zulkarnaen a.s. diminta untuk membangun sebuah dinding yang tinggi dan tebal sehingga tidak dapat ditembus oleh Yakjuj dan Makjuj dan bertujuan untuk melindungi kaumnya dari kejahatan dan kerusakan yang dilakukan oleh mereka. Nabi Zulkarnaen a.s. kemudian memiliki ide untuk membangun sebuah dinding pertahanan yang terbuat dari bahan tembaga dan besi yang panas.

Ternyata, konsep dinding tembaga dan besi panas tersebut diadopsi dalam keamanan teknologi modern yang disebut dengan dinding api (*firewall*) fungsi dari *firewall* yakni untuk menghalau akses dari pihak-pihak yang tidak dikehendaki dan tidak bertanggung jawab terhadap data atau komputer yang dimiliki oleh seseorang.

4.3.2 KEAMANAN DATA

Keamanan *database* tidak hanya berkenaan dengan data yang ada pada *database* saja, tetapi juga meliputi bagian lain dari *system database*, yang tentunya dapat mempengaruhi *database* tersebut. Hal ini berarti keamanan *database* mencakup perangkat keras, perangkat lunak, orang dan data. Agar memiliki suatu keamanan yang efektif dibutuhkan kontrol yang tepat. Seseorang yang mempunyai hak untuk mengontrol dan mengatur *database* biasanya disebut Administrator *database.* Dalam proses pengamanan *database*, ada beberapa aspek yang

harus diperhatikan seorang Administrator dalam upaya mengamankan *database*. Aspek-aspek tersebut seringkali dilupakan oleh para *admin database* sehingga membuat *database* yang dikelolanya menjadi tidak aman lagi. Berikut adalah beberapa hal atau aspek yang harus diperhatikan dalam upaya pengamanan *database*, secara garis besar ada 4 hal yaitu:

a. **KEAMANAN SERVER**

   Perlindungan Server adalah suatu proses pembatasan akses yang sebenarnya pada *database* dalam server itu sendiri. Menurut Blake Wiedman ini adalah suatu sisi keamanan yang sangat penting dan harus direncanakan secara hati-hati. Ide dasarnya adalah kita tidak dapat mengakses apa yang kita tidak dapat lihat, atau apakah kita ingin *database* server kita dapat dilihat di seluruh dunia? *Database* kita bukanlah suatu web server, koneksi yang tidak dikenali tidak diujikan.

b. **TRUSTED *IP ACCESS***

   Setiap server harus dapat mengkonfigurasikan alamat IP yang diperbolehkan mengakses dirinya. Kita tidak mengizinkan semua orang dapat mengakses server kita sebagaimana kita tidak mengizinkan orang lain memasuki rumah kita tanpa izin. Jika server melayani suatu web server maka hanya alamat web server itu saja yang dapat mengakses server *database* tersebut. Jika server *database* melayani jaringan internal maka hanya alamat jaringan lah yang boleh menghubungi server. Sangat perlu diperhatikan

bahwa jangan pernah menggabungkan server *database web* dengan server *database* informasi internal perusahaan anda, ini adalah suatu mental yang buruk untuk seorang *admin. Trusted Ip Access* merupakan server *database* terbatas yang hanya akan memberi respons pada IP yang dikenali saja.

c. **KONEKSI DATABASE**

Saat ini semakin banyaknya aplikasi dinamis menjadi sangat menggoda untuk melakukan akses yang cepat bahkan *update* yang langsung tanpa autentifikasi. Jangan pernah berpikir demikian, ini hanya untuk seorang pemalas. Jika kita ingin mengizinkan pemakai dapat mengubah *database* melalui *web page,* pastikan anda memvalidasi semua masukan untuk memastikan bahwa inputan benar, terjamin dan aman. Sebagai contoh, pastikan anda menghilangkan semua *code* SQL agar tidak dapat dimasukan oleh *user.* Jika anda seorang *admin* yang membutuhkan koneksi ODBC, pastikan koneksi yang digunakan unik.

d. **KONTROL AKSES TABEL**

Kontrol akses tabel ini adalah salah satu bentuk keamanan *database* yang sering diabaikan, karena cukup sulit penerapannya. Penggunaan kontrol akses tabel yang benar dibutuhkan kolaborasi antara sistem administrator dengan pengembang *database.* Hal inilah yang sulit dilakukan. Pemberian izin *user* untuk mengakses informasi dapat membuat informasi terbuka kepada publik. Jika seorang *user* mengakses

informasi apakah akan dilihat menggunakan sesi yang sama? Atau jika tabel digunakan sebagai referensi sistem mengapa ia diberikan izin selain hak membaca saja.

4.3.3 INTEGRITAS DATA

Dilakukan melalui:

a. Pendefinisian struktur tabel dengan membuat indeks primer yang bersifat unik
b. Pengkodean di dalam aplikasi pada saat pemasukan/ penambahan data lebih *user-friendly*
c. Kedua cara diterapkan bersama-sama

4.3.4 KEBOCORAN DATA

Kebocoran data mengacu pada situasi di mana data sensitif secara tidak sengaja terekspos atau diakses oleh pihak yang tidak berwenang. Ancaman dapat terjadi melalui situs web, email, *hard drive,* atau laptop. Perlu Anda ketahui bahwa pembobolan data memiliki arti yang berbeda dengan kebocoran data. Inilah perbedaan antara keduanya, Pembobolan data adalah serangan yang disengaja yang dapat menembus sistem sehingga data sensitif dapat diakses. Kebocoran data tidak memerlukan serangan jaringan khusus, karena biasanya kebocoran data dapat terjadi karena keamanan data yang buruk atau kelalaian pengguna sendiri. Saat terjadi kebocoran data, peretas akan mencuri data sensitif tersebut. Beberapa dari mereka adalah:

1. Informasi identifikasi: nama, alamat, nomor telepon, alamat email, nama pengguna, kata sandi, dll.

2. Aktivitas pengguna: riwayat pemesanan dan pembayaran, kebiasaan *browsing*, dll.
3. Informasi kartu kredit: nomor kartu, tanggal kedaluwarsa, kode pos penagihan, dll.
4. Selain mencari informasi pengguna, peretas juga akan mencuri informasi rahasia milik perusahaan, seperti email, komunikasi internal perusahaan, strategi perusahaan, dll.

4.4. RANGKUMAN

Perkembangan teknologi digital tidak bisa kita hindari, penggunaan teknologi sudah menjadi bagian dari kehidupan kita. Mulai dari belajar, bermain, bahkan berbelanja dan menemukan jodoh pun kita menggunakan teknologi digital. Selain kita mahir dalam menggunakan produk digital, kita perlu juga mahir dalam kemampuan literasi keamanan digital. Seiring perkembangan teknologi digital, berkembang pula dampak negatif dari perkembangan teknologi tersebut. Seperti pencurian data diri, penipuan, peretasan komputer baik pribadi maupun organisasi yang semakin hari semakin marak terjadi. Oleh karena itu kesadaran akan efek negatif tersebut atau yang disebut dengan literasi keamanan digital harus dimiliki oleh setiap pengguna teknologi digital agar tidak terkena dampak negatif atau minimal mengurangi angka korban dari dampak negatif teknologi digital.

4.5 LATIHAN SOAL

Sebagai Latihan untuk mencapai kompetensi yang diharapkan, silakan kerjakan beberapa soal di bawah ini dengan jawaban yang tepat:

1. Apakah itu keamanan data?
2. Sebutkan tipe keamanan data!
3. Mengapa keamanan data itu penting?

BAB 5
IMPLEMENTASI BASIS DATA

Pokok Bahasan
- Pengkodean/Abstraksi
- DBMS dan Struktur Data
- Indeks dan Struktur Tabel
- Struktur Penyimpanan

5.1 PENDAHULUAN

Tahap implementasi Basis Data merupakan upaya untuk membangun Basis Data fisik yang ditempatkan dalam media penyimpan (*disk*) dengan bantuan DBMS. Tahap ini diawali dengan melakukan transformasi dari model data yang telah selesai dibuat struktur Basis Data sesuai DBMS yang dipilih. Secara umum, sebuah ERD akan diwujudkan menjadi sebuah **Basis Data** secara fisik. Sedangkan komponen ER yang berupa himpunan entitas dan himpunan relasi akan diwujudkan menjadi **tabel-tabel**. Atribut-atribut yang melekat pada masing-masing himpunan entitas dan himpunan relasi akan dinyatakan sebagai ***field*** dari tabel yang sesuai.

5.2 TUJUAN PEMBELAJARAN

Setelah mempelajari bab ini, Anda diharapkan mampu:

1. CPMK

Melakukan pemodelan data dan mendemonstrasikan penggunaan SQL secara interaktif untuk melakukan berbagai operasi terhadap Basis Data

2. **SUB CPMK**

Mampu menjelaskan konsep pemodelan data, ER Diagram, UML, Agregasi data, bentuk normalisasi data, penyimpangan dalam data, dalam pengembangan suatu program

5.3 PENGKODEAN/ABSTRAKSI DATA

Data yang dilihat oleh pemakai awam (*end-user*) bisa berbeda dengan bagaimana data/informasi itu disimpan. Apa yang dilihat oleh *end-user* bisa jadi merupakan hasil pengolahan yang tidak disimpan sama sekali dalam Basis Data, atau bisa dinyatakan dalam bentuk lain. Alasan untuk membuat suatu pengkodean adalah untuk efisiensi ruang penyimpanan Dari pemakaiannya, ada dua bentuk pengkodean:

1. Eksternal (*user-defined coding*)

Mewakili pengkodean yang telah digunakan secara terbuka dan dikenal denganbaik oleh pemakai awam. Contoh: Nomor mahasiswa dan Kode matakuliah sudah dikenal baik oleh pemakai awam

a. Internal (sistem *coding*)

Menggambarkan bagaimana data disimpan dalam kondisi sebenarnya, sehingga lebih berorientasi pada mesin. Ada tiga bentuk pengkodean:

1) Sekuensial

Pengkodean dilakukan dengan mengasosiasikan data dengan kode yang urut Contoh: predikat kelulusan “Sangat Memuaskan”, “Cukup Memuaskan”, “Memuaskan” dikodekan dengan huruf “A”, “B”, “C”

2) Mnemonic

Pengkodean dilakukan dengan membentuk suatu singkatan dari data yang hendak dikodekan. Contoh: “Laki-laki” dikodekan ‘L”; “Perempuan” dikodekan “P”

3) Blok

Pengkodean dinyatakan dalam format tertentu. Contoh: Nomor mahasiswa dengan format XX.YY.ZZZZ terdiri atas XX = 2 digit tahun masuk, YY = 2 digit kode jurusan, ZZZZ = 4 digit nomor urut

b. Transformasi Model data ke Basis Data fisik

Aturan umum dalam pemetaan model data yang digambarkan dalam ERD (level konseptual) menjadi Basis Data fisik (level fisik) adalah Setiap himpunan entitas diimplementasikan sebagai sebuah tabel (*file* data).

5.4 DBMS DAN STRUKTUR TABEL

Dalam menentukan struktur dari tabel, paling tidak setiap struktur tabel berisikan nama *field, tipe field*, dan ukurannya. Tatacara penamaan *field,* pilihan *tipe field* serta fasilitas tambahan lainnya untuk struktur tabel sangat tergantung pada DBMS yang digunakan

Tipe data yang bersifat umum adalah:

a. Data *Alphanumerik,* isinya berupa angka tapi tidak menunjukkan jumlah, sehingga dianggap sebagai teks. Misalnya: Nomhs, NIP
b. Data Numerik, isinya berupa angka yang menunjukkan jumlah. Misalnya: SKS,gaji pokok.
c. Data Bilangan Bulat (*integer*), *Byte* (*1 byte*), *Small-integer* (*2 byte*), *Long integer*(*4 byte*)
d. Data bilangan nyata, single (*4 byte*), *Double* (*8 byte*). Tipe data *single* dapat menampung hingga 7 digit pecahan,

sedangkan *double* hingga 15 digit pecahan.

e. Dalam komputasi, data *integer* akan membutuhkan waktu lebih cepat dalam pengolahan data dibandingkan real. Begitu juga, karena ruang penyimpanan yang dibutuhkan lebih kecil, maka data single akan lebih cepat dalam pengolahan dibandingkan *double.*

f. **Data uang (*currency*)**, pemakaian tipe ini sangat membantu dalam mengatur tampilan data yang berkaitan dengan nilai uang, misalnya dengan adanya pemisahan ribuan/jutaan dan adanya tanda mata uang

g. **Data teks**, ada dua jenis yaitu ukuran tetap (*fixed character*) dan ukuran dinamis (*variable character*). Misalnya *fieldnomhs* lebih tepat bertipe *fixed character* karena ukurannya pasti dan pendek. Sedangkan nama mahasiswa sebaiknya bertipe *variable character* karena panjang dan bervariasi.

Pertimbangan dalam menentukan tipe data bagi setiap *field* adalah:

1. Kecukupan domain
   Harus dapat menjamin bahwa tipe data yang dipilih pada tiap *field* akan dapat menampung semua nilai yang akan diisikan ke dalam *field* tersebut.
2. Efisiensi ruang penyimpanan
   Apabila pemilihan tipe data tidak tepat (berlebihan), akibatnya akan memperbesar ukuran tabel secara keseluruhan.
3. Kecepatan pengolahan data
   Pada akhirnya, pemilihan tipe data yang tidak tepat juga mengakibatkan pengaksesan data menjadi lebih lambat.

5.5 INDEKS DAN STRUKTUR PENYIMPANAN

Pada tahap implementasi, atribut-atribut entitas/relasi yang ditetapkan sebagai kunci (*key*) akan diwujudkan sebagai Indeks

Primer (*Primary index*). Dan dapat juga ditambahkan *Secondary index* Ada 2 indeks:

1. Indeks Primer (*primary index*)

   IP pada setiap tabel hanya ada satu dan hampir selalu berasal (ditentukan) dari kunci primer yang telah ditetapkan dalam sebuah entitas/relasi IP yang baik terdiri atas *field-field* dengan kriteria sbb:

   a. Field yang menjadi komponen IP harus bersifat *mandatory* (datanya tidak boleh kosong atau berisi nilai *null*)
   b. Keseluruhan nilai IP bersifat unik
   c. Nilai-nilainya lebih permanen (idealnya tidak pernah berubah)
   d. Berukuran kecil (pendek) dengan jumlah *field* minimal (sedikit)

2. Indeks sekunder (*secondary index*)

   Digunakan untuk mendukung keberadaan IP yang dibuat untuk suatu tabel dengan alasan untuk mempermudah berbagai cara pengaksesan ke suatu tabel. Misalnya: *field* Nama_Mahasiswa untuk memudahkan pencarian data berdasar nama mahasiswa; di samping pencarian berdasar NOMHS. Catatan:

   a. Jumlah IS dalam sebuah tabel boleh lebih dari Satu
   b. Nilai-nilai *field* yang menjadi pembentuk IS tidak harus bersifat unik

5.6 STRUKTUR PENYIMPANAN

Ada 7 pilihan struktur penyimpanan dasar yang dapat diterapkan pada suatu Tabel (bergantung pada DBMS yang dipakai) yaitu: *pile, heap,* hash, sekuensial, berindeks, *file* berindeks, multiring.

a. Heap

Merupakan struktur penyimpanan yang paling sederhana dan paling hemat dalam kebutuhan ruang penyimpanan. Setiap baris data disusun berdasar kronologis penyimpanannya. *Record* yang pertama disimpan akan ditempatkan di posisi awal ruang penyimpanan, dan begitu seterusnya. Pengubahan data tidak akan mengubah urutan *record* tersebut. Jika terjadi penghapusan, maka *record-record* di bawahnya akan dimampatkan untuk mengisi tempat yang kosong akibat penghapusan. Pencarian data berjalan dengan lambat, karena dilakukan secara sekuensial baris demi baris. Struktur ini cocok untuk tabel berukuran kecil dan jarang berubah

b. Hash

Baris-baris data ditempatkan berdasar nilai alamat fisik yang diperoleh dari hasil perhitungan (fungsi *hashing*) terhadap nilai *key*-nya. Karena itu penempatan *record* dalam tabel tidak tersusun berdasarkan kedatangannya. Bisa jadi *record* yang terakhir dimasukkan justru menempati urutan pertama. Memiliki performansi yang paling baik dalam hal pencarian data tunggal berdasar kunci indeks. Struktur ini cocok untuk tabel-tabel yang sering menjadi acuan bagi tabel lain. Kelemahannya membutuhkan ruang penyimpanan awal yang besar, untuk menjamin agar *record-record* yang disimpan tidak menempati alamat yang sama dibutuhkan alokasi ruang penyimpanan

c. Sekuensial berindeks

Menempatkan data dengan urutan tertentu berdasar nilai indeks primernya *Record* yang memiliki nilai IP paling kecil dibandingkan *record* yang lain akan ditempatkandi awal ruang penyimpanan tabel meskipun dimasukkan belakangan. Performansi turun pada saat terjadi penambahan atau perubahan data yang menyangkut nilai indeks primernya, karena perlu dilakukan penataan ulang. Struktur ini cocok

untuk tabel yang sifatnya statis, dan untuk pencarian data kelompok dalam suatu tabel (lebih baik daripada hash)

d. *File* berindeks

Dikembangkan dari struktur *heap*. *Record-record* disusun berdasar kronologis penyimpanannya (seperti *heap*). Namun disediakan pula *file* indeks yang disusun berdasar nilai *key* setiap *record* yang berguna untuk membantu proses pencarian data ke suatu *table.* Terdapat 2 komponen yaitu komponen data dan komponen indeks. Komponen data disusun dengan struktur *heap,* dan komponen indeks disusun dengan struktur sekuensial berindeks. Struktur ini cocok untuk tabel yang dinamis dan berukuran besar.

5.7 RANGKUMAN

Implementasi Basis Data merupakan suatu **tahapan dalam proses perancangan Basis Data**. Tahap ini merupakan implementasi dari hasil pemodelan *logical* dan fisikal. Bahasa perintah yang digunakan, baik itu untuk definisi data ataupun penyimpanan data harus sesuai dengan DBMS yang dipilih.

5.8 LATIHAN SOAL

Sebagai Latihan untuk mencapai kompetensi yang diharapkan, silakan kerjakan beberapa soal di bawah ini dengan jawaban yang tepat:

a. Sebutkan pengkodean dalam sebuah sistem!

b. Tipe *database* adalah?

BAB 6
NORMALISASI

Pokok Bahasan
- Penyimpangan dalam Modifikasi
- Keharusan Menghilangkan Masalah-Masalah Akibat Ketergantungan
- Efek-Efek Normalisasi
- Domain dan Tipe Data
- Bentuk-Bentuk Normal

6.1. PENDAHULUAN

Normalisasi merupakan sebuah teknik *logical design* dalam sebuah Basis Data yang mengelompokkan atribut dari berbagai entitas dalam suatu relasi sehingga membentuk struktur relasi yang baik (tanpa redudansi/pengulangan data) serta sebagian besar *ambiguity* bisa dihilangkan atau pengertian singkatnya, normalisasi *database* adalah proses pengelompokan atribut data yang membentuk entitas sederhana, non redundan, fleksibel, dan mudah beradaptasi, Sehingga dapat dipastikan bahwa *database* yang dibuat berkualitas baik. Normalisasi *database* terdiri dari banyak bentuk, dalam ilmu Basis Data ada setidaknya 9 bentuk normalisasi yang ada yaitu 1NF, 2NF, 3NF, EKNF, BCNF, 4NF, 5NF, DKNF, dan 6NF. *Database* 1NF, 2NF, dan 3NF akan sering ditemui ketika akan membuat sebuah *database* yang optimal. Jika Anda ingin menjadi seorang *Database* Administrator (DBA), harus tahu bagaimana cara normalisasi *database* yang optimal. Misalkan suatu saat ketika *website* yang Anda buat mengalami penurunan kinerja, mungkin Anda akan ditanya apakah *database* tersebut sudah di normalisasi dengan benar.

6.2. TUJUAN PEMBELAJARAN

Setelah mempelajari bab ini, Anda diharapkan mampu:

1. CPMK

Melakukan pemodelan data dan mendemonstrasikan penggunaan SQL secara interaktif untuk melakukan berbagai operasi terhadap Basis Data.

2. SUB CPMK

Mampu menjelaskan konsep pemodelan data, ER Diagram, UML, Agregasi data, bentuk normalisasi data, penyimpangan dalam data, dalam pengembangan suatu program.

6.3. PENGERTIAN

Normalisasi adalah suatu teknik yang menstrukturkan data dalam cara-cara tertentu untuk membantu mengurangi atau mencegah timbulnya masalah yang berhubungan dengan pengolahan data dalam Basis Data. Kriteria yang mendefinisikan level-level pada normalisasi adalah bentuk normal (*norm form*). Normalisasi adalah sebuah metode yang urut dalam penerapan aturan-aturan untuk mendesain *database* dengan tujuan agar (tabel menjadi normal):

6.3.1.1. Meminimalkan *redundancy* dan pengulangan data

6.3.1.2. Mempertahankan integritas data

6.3.1.3. Menambah konsistensi dan stabilitas

6.3.1.4. Menghilangkan potensi *anomaly* ketika mengolah data, anomali yang dimaksud:

   a. insertion anomalies, deletion anomalies, update anomalies.

   b. Level-level dalam normalisasi disebut *Normal Form* (NF)

   c. Penggagas pertama: Edgar F. Codd

6.4. TUJUAN NORMALISASI

Normalisasi perlu dilakukan agar kerelasian dalam Basis Data menjadi mudah dimengerti, mudah dipelihara, mudah memprosesnya, dan mudah untuk dikembangkan sesuai kebutuhan baru. Tahapan-tahapan normalisasi

6.4.1.1. Bentuk Normal tahap Pertama (*1st Normal Form/1NF*)
6.4.1.2. Bentuk Normal tahap Kedua (*2nd Normal Form/2NF*)
6.4.1.3. Bentuk Normal tahap Ketiga (*3rd Normal Form/3NF*)
6.4.1.4. Bentuk Normal tahap Keempat (*4th Normal Form/4NF*)
6.4.1.5. Bentuk Normal tahap Kelima (*5th Normal Form/5NF*)

6.5. PENYIMPANGAN DALAM MODIFIKASI

1. Penyimpangan dalam proses modifikasi data disebut *anomalies*
2. Ada 3 bentuk penyimpangan:
   a. *Delete anomalies*

   Adalah proses penghapusan suatu *entity* logik yang mengakibatkan hilangnya informasi tentang *entity* yang tidak direlasikan secara logik.

Tabel 7.1. Tabel Kuliah

No. Mhs	Nama	Kode Mtk	SKS
123456	Ali Baba	INA 101	3
123457	Pipiyot	TFD 234	2
123467	Nirmala	INA 201	3
123445	Lala	INA 101	3

Apabila “Ali Baba” membatalkan mengambil matakuliah “INA 101”, maka apabila *record* tersebut dihapus akan menyebabkan seluruh informasi tentang ‘Ali baba” akan ikut terhapus.

b. *Insert anomalies*

Adalah proses penyisipan *entity logic* yang memerlukan

penyisipan entity logik yang lain.

c. *Update anomalies*

Adalah proses meng-*update* data pada suatu *entity* logik yang mengakibatkan perubahan pada lebih dari satu tempat dalam suatu relasi.

Contoh: Perubahan SKS pada "INA 101" tidak hanya dilakukan pada satu *record* saja, tetapi pada *record* dan relasi lain yang memuat data tersebut.

6.6. KEHARUSAN MENGHILANGKAN MASALAH-MASALAH AKIBAT KETERGANTUNGAN

Yang harus dilakukan adalah jika struktur data dalam relasi dirancang sedemikian rupa sehingga atribut-atribut bukan kunci hanya tergantung pada atribut kunci dan tidak pada atribut lain:

Ada 3 ketergantungan:

a. *Functional Dependence* (FD)

FD akan muncul di antara dua rinci data dalam suatu struktur data jika nilai salah satu rinci data mengimplikasikan nilai pada rinci data kedua atau rinci data pertama menentukan (*determines*) rinci data kedua.

Contoh:

Matakuliah (Kode, Nama, SKS, Semester)

FD=Matakuliah.Kode→(Matakuliah.Nama,Matakuliah.Semester) Matakuliah.nama →((Matakuliah. Kode, Matakuliah. Semester)

b. *Full Functional Dependence* (FFD)

Suatu rinci data dikatakan FFD pada suatu kombinasi rinci data jika FD pada kombinasi rinci data dan tidak FD pada bagian lain dari kombinasi rinci data

Contoh: SKS pada tabel matakuliah hanya bergantung pada kode matakuliah, dan tidak ditentukan oleh siapa yang mengambil matakuliah tersebut.

c. *Transitive Dependence* (TD)

Muncul jika suatu nilai pada rinci data pertama menentukan nilai pada rinci data kedua yang bukan CK, dan nilai pada rinci data kedua menentukan nilai pada rinci data ketiga jadi TD terjadi jika suatu nilai rinci data mempunyai ketergantungan dengan pada dua nilai rinci data.

6.7. EFEK-EFEK NORMALISASI

Akibat yang muncul dalam proses normalisasi:

a. Masalah kekangan dalam Basis Data yaitu duplikasi rinci data dan adanya Integritas referensial yang harus terjaga dan nilai-nilai pada AK tidak boleh *null* maka proses dekomposisi akan menghasilkan suatu set yang inheren pada batasan integritas referensial.

b. Ketidakefisienan dalam menampilkan kembali data tersebut.

6.8. ATRIBUT TABEL

Atribut adalah karakteristik atau sifat yang melekat pada sebuah tabel, atau disebut jugakolom data. Pengelompokan atribut:

a) Atribut Key

Adalah satu atau gabungan dari beberapa atribut yang dapat membedakan semua baris data dalam tabel secara unik (tidak boleh ada dua atau lebih baris data dengan nilai yang sama untuk atribut tertentu). Ada 3 *key* dalam *database:*

1) Superkey

a) Merupakan satu atau kumpulan atribut yang dapat membedakan setiap baris data dalam sebuah tabel secara unik

Contoh: *superkey* di tabel mahasiswa

(no mhs, nama, alamat, tgl lahir)
(nomhs, nama, tgl lahir)
(nomhs, nama)

(nomhs)

b) *Candidate key*

Merupakan kumpulan atribut minimal yang dapat membedakan setiap baris data dalam sebuah tabel secara unik. *Candidate key* adalah satu atau lebih kolom yang nilainya dapat digunakan untuk memberi identitas unik sebuah baris dalam tabel. Sebuah CK pasti *superkey*, tapi belum tentu sebaliknya.

Contoh: pada tabel mahasiswa

(nomhs)

(nama)

c) *Primary key*

Sebuah *Primary Key* adalah satu atau lebih kolom yang nilainya untukmemberi identitas unik sebuah baris dalam tabel. *Primary key* dipilih di antara *candidate key* yang ada.

Dari beberapa CK dapat dipilih satu untuk dijadikan PK, yang memiliki keunikan paling baik.

Contoh: dari tabel mahasiswa, yang layak dijadikan PK adalah no.mhs.

d) Atribut deskriptif

Merupakan atribut yang bukan merupakan anggota dari PK.

e) Atribut sederhana

Adalah atribut atomik yang tidak dapat dipilah lagi.

Contoh: Nomhs, Nama

f) Atribut Komposit

Adalah atribut yang masih bisa diuraikan lagi menjadi sub-atribut yang masing- masing memiliki makna.

Contoh: Alamat□Alamat, Kota, Propinsi, Kode Pos

g) Atribut bernilai tunggal

Ditujukan pada atribut-atribut yang memiliki paling banyak satu nilai untuk setiapbaris data.

Contoh: Nomhs, Nama, Tanggal lahir □hanya dapat berisi satu nilai untuk seorang mahasiswa

h) Atibut bernilai banyak

Ditujukan pada atribut-atribut yang dapat diisi dengan lebih dari satu nilai, tapi jenisnya sama.

Contoh: pada tabel mahasiswa dapat ditambah atribut HOBBY, karena seorang mahasiswa dapat memiliki beberapa hobby

i) Atribut harus bernilai (*mandatory*)

Adalah atribut yang tidak boleh kosong, atau harus ada nilainya. Misalnya data Nomhs dan Nama mahasiswa.

Nilai NULL digunakan untuk mengisi atribut yang demikian yang nilainya belumsiap atau tidak ada.

NULL (karakter ke 0) tidak sama dengan SPASI (karakter ke 32).

6.8. DOMAIN DAN TIPE DATA

1. Domain, memiliki pengertian yang hampir sama dengan tipe data, namun domain lebih ditekankan pada batas-batas nilai yang diperbolehkan pada suatu atribut
2. Contoh: data SKS bertipe *integer*. Namun dalam kenyataan tidak ada SKS yang bernilai negatif. Berarti domain nilai SKS adalah integer > 0
3. Tipe data merujuk pada kemampuan penyimpanan data yang mungkin bagi suatu atribut secara fisik a kenyataan pemakaian

6.9 BENTUK-BENTUK NORMAL

Normalisasi merupakan sebuah teknik dalam *logical* desain sebuah Basis Data, teknik ukur struktur relasi yang baik (tanpa redundansi)

6.8.1.1. Normal pertama (*1st normal form*)

Syarat:

a. Tidak ada atribut yang duplikat dalam sebuah tabel.
b. Tidak ada baris yang duplikat dalam sebuah tabel.
c. Nilai *cell* dalam harus *atomic value* (*single value*).
d. Tidak ada pengulangan *group data* (pengulangan item di kolom).

Langkah:

a. Hilangkan atribut yang duplikat.
b. Buatlah menjadi tabel terpisah untuk masih-masing group data dan buatatribut relasinya (jika ada).
c. Identifikasi setiap set relasi data dengan satu atau beberapa kolom unik (*primary key*).

Aturan:

a. Mendefinisikan atribut kunci
b. Tidak adanya grup berulang
c. Semua atribut bukan kunci tergantung pada atribut kunci

6.8.1.2. Normal kedua (*2nd normal form*)

Syarat:

a. Sudah memenuhi 1NF
b. Atribut *non-key* secara fungsi tergantung penuh pada *primary key*
c. Tidak ada *partial dependencies*: tidak ada atribut yang tergantung pada sebagian dari *primary key* (untuk kasus *composite primary key*)

Langkah:

a. Jika ada atribut yang tergantung pada sebagian *primary key*, pecah menjadi tabel sendiri atau cari data yang terulang kemudian pecah menjadi tabel sendiri.
b. Kemudian buat relasi di antara set data yang dipisahkan.

Aturan:

a. Sudah memenuhi bentuk normal pertama
b. Tidak ada ketergantungan parsial (di mana seluruh *field* hanya tergantung padasebagian *field* kunci)

6.8.1.3. Normal ketiga (*3rd normal form*)

Syarat:

a. Memenuhi 2NF
b. Tidak ada atribut yang tergantung secara transitif pada non-*key* lainnya

Langkah:

a. Hapus atau pisahkan menjadi tabel sendiri atribut yang tergantung pada kolom *non-key* (biasanya pada atribut turunan).
b. Pastikan semua atribut non-*key* tergantung pada *primary key.*

Aturan:

a. Sudah berada dalam bentuk normal kedua
b. Tidak ada ketergantungan transitif (di mana *field* bukan kunci tergantung pada *field* bukan kunci lainnya)

6.8.1.4. Normal Boyce-Codd (*Boyce Codd Norm Form*)

a. Sebuah tabel dikatakan memenuhi BCNF jika untuk semua dengan notasi X ->Y, maka X harus merupakan *super key* pada tabel tersebut.
b. Jika belum demikian maka tabel tersebut harus di dekomposisikan ulang berdasar KF yang ada,

sedemikian hingga X menjadi *super key* dari tabel hasil dekomposisi.

c. Sebuah tabel dikatakan memenuhi bentuk normal ketiga (3NF), jika untuk setiap ketergantungan fungsional dengan notasi X -> A, di mana A mewakili semua atribut tunggal di dalam tabel yang tidak ada di dalam X, maka: X haruslah *super key* pada tabel tersebut dan A merupakan bagian dari *primary key* pada tabel tersebut.

Aturan:

a. Sudah berada dalam bentuk normal ketiga
b. Semua determinannya merupakan *candidate key*

6.8.1.5. Proses Normalisasi

Catatan:

Bentuk normal seharusnya berada dalam bentuk normal tertinggi dan bergerak dari bentuk normal pertama dan seterusnya untuk setiap kali membatasi hanya satu jenis redundansi.

Keseluruhan ada 5 bentuk normal. Tiga bentuk normal pertama menekankan redundansi muncul dari *Functional Dependencies* sedangkan bentuk keempat dan kelima menekankan redundansi yang muncul dari kasus *Multi Valued Dependencies.*

6.10 LATIHAN SOAL

Tugas Kelompok

1. Terdapat relasi yang belum di normalisasi sebagai berikut.

NO_TRAN	BARANG	JUMLAH	HARGA_UNIT	PELANGGAN	TELPON

A0001	Gula Pasir Minyak	1 2	300.000 280.000	Adil	589125
A0002	Beras	3	225.000	Formula	123456
A0003	Gula Pasir	1	310.000	Adil	589125
A0004	Kacang	1	160.000	Setia	144567

a) Bagaimana bentuk normal pertamanya?

b) Bagaimana komposisi relasi-relasi setelah bentuk normal kedua diterapkan?

2. Sebuah relasi beserta isinya adalah sebagai berikut: MAHASISWA(NIM, NAMA, DOSEN_WALI)
NIM = nomor induk mahasiswaNAMA = nama mahasiswa
DOSEN_WALI = nama dosen wali (pembimbing akademis)
Seorang mahasiswa hanya memiliki seorang dosen wali.

a) Apakah DOSEN_WALI mempunyai dependensi fungsional terhadap NIM?

b) Apakah NIM mempunyai dependensi fungsional terhadap DOSEN_WALI?

3. Relasi PROYEK berupa:
PROYEK (KODE_PROYEK, NAMA_PEGAWAI, BAGIAN)
Sampel data:

KODE_PROYEK	NAMA_PEGAWAI	BAGIAN
A203	Amir	EDP
A203	Udin	HRD
A203	Wawan	HRD
A204	Amir	EDP
A204	Fika	Akunting
A205	Amir	EDP
A205	Wawan	HRD
A205	Fika	Akunting

M
enurut Anda, pertanyaan-pertanyaan manakah yang benar?

a) KODE_PROYEK → NAMA_PEGAWAI

b) KODE_PROYEK → BAGIAN

c) (KODE_PROYEK, NAMA_PEGAWAI) → BAGIAN

d) NAMA_PEGAWAI → BAGIAN
e) BAGIAN → KODE_PROYEK
f) BAGIAN → NAMA_PEGAWAI
g) BAGIAN → (KODE_PROYEK, NAMA_PEGAWAI)

4. Berdasarkan relasi PROYEK pada soal di atas:
   a) Apakah PROYEK memenuhi bentuk normal pertama? Jelaskan!
   b) Apakah PROYEK memenuhi bentuk normal kedua? Jelaskan!
   c) Apakah PROYEK memenuhi bentuk normal ketiga? Jelaskan!
   d) Anomali apa saja yang mungkin terjadi pada relasi tersebut (dengan memberikancontoh)?
   e) Apa kunci primer bagi relasi PROYEK?
   f) Apakah pada relasi tersebut terdapat dependensi transitif? jika ada, sebutkan!
   g) Bagaimana bentuk relasinya agar persoalan-persoalan anomali dapat dihilangkan?

BAB 7
SQL

Pokok Bahasan
- Struktur Dasar
- Struktur Select
- Klausa Select
- Klausa Form
- Klausa Where
- Limit & Offset
- Fungsi Agregasi
- DDL

7.1 PENDAHULUAN

DBMS umumnya menyediakan program khusus (utilitas/*utility*) yang dapat digunakan secara interaktif untuk melakukan berbagai operasi terhadap Basis Data, seperti pembuatan tabel, penghapusan tabel, penambahan data, pengubahan data, pencarian data, penghapusan data dan lain-lain. Namun di samping adanya program khusus itu, DBMS juga umumnya menyediakan sekumpulan perintah (dalam bentuk *commandline*, yakni perintah yang dituliskan pemakai) untuk maksud yang sama. Perintah-perintah ini dapat diberikan (dan dikerjakan oleh DBMS) melalui utilitas lain yang juga disediakan DBMS atau melalui program/aplikasi yang dibuat sendiri oleh pemakai. Kumpulan perintah ini dapat disebut sebagai Bahasa Basis Data (*Database Language*).

Ada banyak sekali bahasa Basis Data yang pernah dibuat untuk masing-masing DBMS. Namun akhirnya yang menjadi standar adalah SQL. SQL merupakan kependekan dari *Structured Query Languange* (Bahasa *Query* yang Terstruktur). Istilah *Query Languange* memang tidak tepat sama dengan istilah Bahasa Basis Data (*Database Languange*). Bahasa Basis Data terdiri atas *Data Definition Language* (DDL) dan *Data Manipulation Language*

(DML).

7.2 TUJUAN PEMBELAJARAN

Setelah mempelajari bab ini, Anda diharapkan mampu:

1. CPMK

Melakukan pemodelan data dan mendemonstrasikan penggunaan SQL secara interaktif untuk melakukan berbagai operasi terhadap Basis Data

2. SUB CPMK

Melakukan perumusan dan mendemonstrasikan fungsi dalam suatu *syntak* dan fungsi SQL serta manipulasi data serta perintah yang digunakan untuk pendefinisian objek Basis Data (DDL)

7.3 STRUKTUR DASAR

Sebuah ekspresi SQL dasar sebenarnya hanya terdiri atas 3 klausa, yaitu: ***select, from*** dan *where*:

1. Klausa ***select*** digunakan untuk menetapkan daftar atribut (*field*) yang diinginkansebagai hasil *query.*
2. Klausa ***from*** digunakan untuk menetapkan tabel (atau gabungan tabel) yang akan ditelusuri selama *query* data dilakukan.
3. Klausa ***where***, yang sifatnya opsional, digunakan sebagai predikat (kriteria) yang harus dipenuhi dalam memperoleh hasil *query*.

Sintaks (cara penulisan) dari ekspresi SQL dasar dengan 3 klausa tersebut:

Select A1 [, A2, … , An]**From** t1 [, t2, … , tm] [**where** P]

Di mana:

- A1, A2, …, An merupakan daftar atribut.
- t1, t2, …, tm merupakan daftar tabel.
- P merupakan predikat *query*.
- [] merupakan tanda opsional (boleh digunakan, boleh tidak

digunakan).

7.4 STRUKTUR *SELECT*

Bentuk lain struktur *select:*

SELECT [ALL | DISTINCT [ON (*expression* [, ...])]]
* | *expression* [AS *output_name*] [, ...][FROM *from_item* [, ...]]

[WHERE condition]

[GROUP BY expression [, ...]]

[HAVING condition [, ...]]

[{ UNION | INTERSECT | EXCEPT } [ALL] select]

[ORDER BY expression [ASC | DESC | USING operator] [, ...]

][FOR UPDATE [OF tablename [, ...]]] [LIMIT { count | ALL

}][OFFSET start]

7.5 KLAUSA SELECT

1. Untuk melihat semua kolom dari suatu tabel: *SELECT* * *FROM* nasabah;
2. Untuk melihat kolom-kolom tertentu: *SELECT* nama_nasabah *FROM* nasabah;
   SELECT id_nasabah, nama_nasabah *FROM* nasabah;
3. Secara umum:
   SELECT <nama kolom,...> *FROM* <nama tabel>;
   AS digunakan untuk mengganti nama kolom pada tampilan *SELECT*. Contoh: *SELECT* nama_nasabah AS "Nama Nasabah"
   FROM nasabah;
   SELECT nama_nasabah AS "Nasabah", alamat_nasabah AS "AlamatNasabah"
   FROM nasabah;

7.6 KLAUSA FROM

Klausa ini digunakan untuk menetapkan tabel yang kita jadikan sebagai sumber (lokasi) pencarian data. Sebagaimana kita ketahui, Basis Data terdiri atas sejumlah tabel yang saling berhubungan. Karena itu, akan seringkali ada kebutuhan untuk melakukan *query* tidak hanya dari satu tabel, tapi dengan merelasikan beberapa tabel sekaligus. Upaya ini dilakukan karena atribut-atribut yang kita harapkan sebagai hasil *query* tidak hanya tersedia di sebuah tabel, tapi berada di sejumlah tabel.

Contoh-contoh sebelumnya hanya menunjukkan *query* terhadap sebuah tabel. Sebagai hasil implementasi, tabel Kuliah terdiri atas atribut-atribut (*field*) *kode_kul, nama_kul, sks, semester* dan *kode_dos*. Jika kita ingin menampilkan data kuliah beserta dosen-dosen yang mengajarkannya, maka kita tidak hanya dapat melakukan *query* dari tabel Kuliah saja, karena data seperti nama dosen tidak tersimpan di tabel ini, tetapi berada di tabel Dosen. Untuk memenuhi keinginan itu, kita dapat menggunakan ekspresi SQL berikut:

select * ***from*** kuliah, dosen
where kuliah.kode_dos = dosen.kode_dos

Perlu diperhatikan, melakukan *query* terhadap 2 tabel atau lebih tidak bisa dilakukan sembarangan. Tabel-tabel yang menjadi sumber *query* harus memiliki keterhubungan (relasi). Pada perintah di atas, keterhubungan itu diwakili oleh kesamaan nilai pada atribut *kode_dos* dan kita tahu bahwa atribut ini dimiliki oleh kedua tabel. Ekspresi ‘kuliah.kode_dos’ menunjukkan nilai *kode_dos* yang berasal dari tabel Kuliah dan ‘dosen.kode_dos’ menunjukkan nilai *kode_dos* yang berasal dari tabel Dosen.

Kita dapat menggunakan nama alias untuk tabel-tabel pada klausa *from* untuk menyederhanakan penulisan. Ekspresi *query* di

atas, dengan hasil yang sama, dapat pula dinyatakan dengan ekspresi sebagai berikut:

select ****from*** kuliah k, dosen d
where k.kode_dos = d.kode_dos

Terhadap sumber data (tabel *query*) yang banyak, tanda * (asterisk) pada klausa *select* akan mengacu pada semua atribut yang ada di semua tabel yang disebutkan pada klausa ***from***. Jika kita hanya ingin menampilkan atribut-atribut tertentu saja, maka nama tabel atau aliasnya dapat kita gunakan untuk memperjelas asal atribut yang kita tampilkan tersebut, misalnya:

select k.kode_kul, k.nama_kul, d.nama_dos
from kuliah k, dosen d
where k.kode_dos = d.kode_dos

7.7 KLAUSA *WHERE*

Digunakan untuk membatasi hasil *SELECT* yang ditampilkan berdasarkan kondisi yangditentukan. Contoh:

SELECT nama_nasabah FROM nasabah WHERE nama_nasabah = 'Ali Topan';

SELECT nama_nasabah, alamat_nasabah FROM nasabah
WHERE id_nasabah = 2;

Bisa menggunakan >, <, <> (atau !=), >=, <=

Gunakan *AND* atau *OR* untuk lebih dari satu kondisi: *SELECT* * *FROM* nasabah
WHERE nama_nasabah = 'Rina Marsudi' *AND* alamat_nasabah = 'Jl. Kusumanegara 30';

SELECT * *FROM* nasabah *WHERE* nama_nasabah = 'Ali Topan' OR id_nasabah = 2;

Pencarian *NULL*

Gunakan IS *NULL* untuk mencari *NULL:*

SELECT * *FROM* rekening *WHERE* kode_cabang IS NULL;

Gunakan *IS NOT NULL* untuk mencari yang tidak *NULL:SELECT * FROM* rekening
WHERE kode_cabang *IS NOT NULL;*
Pencarian *String*

1) Gunakan LIKE untuk mencari *string* tertentu: *SELECT * FROM* nasabah
   WHERE nama_nasabah LIKE 'Ali Topan';
2) Bisa menggunakan %, berarti cocok untuk semua *substring*
   *SELECT * FROM* nasabah
   WHERE alamat_nasabah LIKE '%negara%';
3) Bisa menggunakan _ untuk 1 huruf. Tanda '_' berarti cocok untuk semuakarakter pada posisi yang sesuai
   SELECT * FROM nasabah
   WHERE nama_nasabah LIKE 'Ali T_p_n';

7.8 *LIMIT & OFFSET*

Digunakan untuk membatasi jumlah baris yang ditampilkan dalam *SELECT*.
Hanya menampilkan 3 baris pertama:
*SELECT * FROM* nasabah*ORDER BY* id_nasabah LIMIT 3;
Menampilkan 2 baris setelah melewati 2 baris pertama: *SELECT * FROM* nasabah

ORDER BY id_nasabahLIMIT 2 OFFSET 2;
Penggunaan *LIMIT* sebaiknya selalu digunakan bersama dengan *ORDER BY*, sehingga urutan yang ditampilkan akan selalu konsisten. *LIMIT* dan *OFFSET* sangat berguna dalam tampilan yang berbasis web (misalnya dengan menggunakan PHP atau JSP) agar tampilan data tidak terlalu besar dan bisa lebih rapi. Tampilan data yang banyak bisa diatur dan dibagi menjadi beberapa halaman (*pages*).

7.9 FUNGSI AGREGASI

Di samping menampilkan nilai-nilai atribut yang ada di dalam tabel, sering pula ada kebutuhan untuk menampilkan data-data agregasi, seperti banyaknya *record*, total nilai atribut, rata-rata nilai atribut, nilai atribut terbesar ataupun nilai atribut terkecil. Data agregasi semacam itu dapat diperoleh dengan menggunakan fungsi-fungsi berikut ini:

count untuk mendapatkan nilai banyaknya *record* hasil *query*,

sum untuk mendapatkan nilai total suatu atribut numerik hasil *query*,

avg untuk mendapatkan nilai rat-rata suatu atribut numerik hasil *query*

max untuk mendapatkan nilai terbesar dari suatu atribut hasil *query*,

min untuk mendapatkan nilai terkecil dari suatu atribut hasil *query*.

Berikut adalah contoh-contoh penggunaan fungsi agregasi tersebut

1. Menampilkan banyaknya *record* mahasiswa:
   select *count* (*)
   from mahasiswa
2. Menampilkan banyaknya mahasiswa angkatan 98:
   select *count* (*)
   from *mahasiswa*
   where *nim* ***like*** ‘98%’

3. Menampilkan total SKS untuk kuliah di semester 2:
   select sum (SKS)
   from kuliah
   where semester = 2

4. Menampilkan rata-rata SKS untuk semua mata kuliah:
   select *avg*(*sks*) ***from*** kuliah
5. Menampilkan indeks nilai terbesar yang diperoleh mahasiswa untuk mata kuliah dengan kode kuliah 'IF-110':
   select *max* (indeks_nilai) ***from*** nilai
   where kode_kul = 'IF-110'
6. Menampilkan tanggal lahir paling tua yang ada di tabel mahasiswa:
   select min(tgl_lahir)
   from mahasiswa

7.10 MANIPULASI DATA

1. Penambahan *Record*

   Sintaks SQL untuk penambahan *record* baru ke sebuah tabel adalah:

   Insert into t [(A1, A2, …, An)] **Values** (v1, v2, …, vn) Di mana:

   a. t adalah nama tabel yang akan mengalami penambahan *record*

   b. A1, A2, …, An adalah nama-nama atribut yang akan diisi nilai

   c. v1, v2, …, vn adalah nilai-nilai yang akan mengisi atribut-atribut tersebut

   Berikut ini adalah contoh perintah SQL untuk melakukan penambahan *record* baru ketabel Mahasiswa:

   Insert into Mahasiswa (nim, nama_mhs, alamat_mhs, tgl_lahir)

   Values ('980011', 'Siti Zubaedah', 'Jl. Guntur Kulon 12, Bogor', '02/03/1973')

   Insert into Mahasiswa

   Values ('980011', 'Siti Zubaedah', 'Jl. Guntur Kulon 12,

Bogor', '02/03/1973')

Jika kita melakukan penambahan *record* baru dengan perintah berikut:

Insert into Mahasiswa (nim, nama_mhs, alamat_mhs)
Values ('980011', 'Siti Zubaedah', 'Jl. Guntur Kulon 12, Bogor')

Maka atribut *tgl_lahir* yang tidak disebutkan dalam perintah ***insert*** tersebut akan diisi dengan nilai *null.*

2. Pengubahan *Record*
   Sintaks SQL untuk pengubahan nilai atribut pada suatu *record* dari sebuah tabel adalah:

7.10 DATA DEFINITION LANGUAGE

Data Definition Language (DDL) berkaitan dengan perintah-perintah untuk pendefinisian objek-objek Basis Data. Salah satu objek terpenting adalah tabel. Berikut ini adalah Sintaks SQL untuk melakukan pembuatan tabel baru di dalam Basis Data:
create table t (A1 D1, A2 D2, … , An Dn) di mana:

a. t adalah nama tabel yang akan dibuat
b. A1, A2, …, An adalah nama-nama atribut yang akan terdapat di dalam tabel t
c. D1, D2, …, Dn adalah domain nilai masing-masing atribut tersebut yang ditentukan berdasarkan tipe datanya

Berikut ini adalah contoh perintah SQL untuk membuat tabel Mahasiswa:

create table mahasiswa
(nim char(6), nama_mhs varchar(30),
alamat_mhs varchar(60),tgl_lahir *date*)

Tabel yang telah dibuat juga dapat dibatalkan keberadaannya

(penghapusan tabel) dengan menggunakan perintah SQL dengan sintaks berikut ini:

drop table t

Contoh penghapusan terhadap tabel Mahasiswa:

drop table mahasiswa

Jika terhadap tabel yang kita buat, ingin kita sertakan pula adanya Indeks Primer berdasarkan atribut tertentu di dalam tabel, maka klausa ***primary key*** dapat kita gunakan. Berikut ini adalah contoh ekspresi SQL untuk pembuatan tabel Mahasiswa sekaligus dengan pendefinisian Indeks Primer berdasarkan *nim*:

create table mahasiswa
(nim char(6), nama_mhs varchar(30),
alamat_mhs varchar(60),tgl_lahir *date*,
primary key (nim))

Jika jumlah atribut yang membentuk Indeks Primer ada lebih dari satu, seperti yang ada di tabel Nilai, contoh ekspresi SQL-nya adalah:

create table nilai
(nim char(6), Kode_kul char(6), Indeks_kul char(1),tgl_lahir date,
primary key (nim, kode_kul))

Struktur sebuah tabel juga dapat kita ubah, tanpa harus menghapus dan kemudian membangunnya kembali dengan definisi struktur yang baru. Perintah pengubahan struktur selain lebih praktis, juga tidak mengakibatkan hilangnya data yang sudah ada di dalam tabel (jika memang sudah terisi data). Perubahan struktur ini dapat berupa penambahan atribut atau pengurangan/penghapusan atribut tertentu. Sintaks SQL untuk perubahan struktur tabel yang berbentuk penambahan atribut baru ke tabel t adalah:

alter table *t* ***add*** *A D*

Di mana t mewakili tabel, A mewakili nama atribut dan D mewakili

tipe data untuk atribut A tersebut.
Sedang untuk penghapusan atribut dari tabel t, sintaks SQL-nya adalah:

alter table t drop A

Berikut ini adalah contoh ekspresi SQL untuk penambahan atribut baru bernama *ip* ditabel Mahasiswa:

alter table mahasiswa ***add*** *ip numeric* (5, 2)

Jika atribut *ip* ingin dihapus dari tabel Mahasiswa, ekspresi SQL-nya:

alter table mahasiswa ***drop*** *ip*

7.11 LATIHAN SOAL

Tabel: tblpengarang				
Kd_peng	Nama	Alamat	Kota	Kelamin
1	Asadi	Jl. Beo 34	Yogya	P
2	Rian H.	Jl. Solo 123	Yogya	P
3	Suadi Marwan	Jl. Semangka II/1	Bandung	P
4	Siti Halimah	Jl. Sukaria 5	Solo	W
5	Amir Hamzah	Jl. Gajah Mada 18A	Kudus	P
6	Suparman	Jl. Setia 1	Jakarta	P
7	Jaja M	Jl. Hangtuah 3	Bandung	P
8	Saman	Jl. Gedong Kuning	Yogya	P
9	Anwar Junaidi	Jl. Tidar 6A	Magelang	P
10	Fatmawati	Jl. Renjana 4	Bogor	W

Tabel: tblbuku		
Kd_buku	Judul	Kd_peng
1	Pemrograman C++	1
2	Pengantar Basis Data	1
3	Panduan Microsoft Office	2
4	Pemrograman Visual dBASE	1

5	Sistem Pakar	4
6	Pemrograman C++	3
7	Visual C++	6
8	QBASIC	5
9	Pemrograman Pascal	8

1. Berdasarkan tabel tblpengarang, tuliskan pernyataan SQL untuk menampilkan
   a) Nama pengarang yang kode pengarangnya adalah 5.
   b) Semua nama pengarang yang tinggal di Yogya.
   c) Semua nama pengarang yang tidak tinggal di Yogya.
   d) Semua nama pengarang berkelamin wanita.
   e) Nama, alamat, dan kota pengarang yang berkelamin pria.
2. Berdasarkan tabel tblbuku, tuliskan pernyataan SQL untuk menampilkan:
   a) Semua judul buku yang ditulis oleh pengarang berkode 1.
   b) Semua data (semua kolom dan semua baris).
3. Tuliskan pernyataan untuk menciptakan tabel bernama tblpegawai, dengan komposisi sebagai berikut:
   NIM bertipe karakter, panjang 6 Nama bertipe karakter, panjang 25Gaji bertipe numerik 8 digit
4. Tuliskan pernyataan untuk membuat indeks dengan nama idx_nip yang dikenakan terhadap kolom NIP pada tabel tblpegawai.
5. Bagaimana cara menghapus indeks idx_nip yang baru saja Anda ciptakan?
6. Bagaimana perintahnya agar panjang Nama (pada tabel tblpegawai) tidak lagi erupa 25karakter, melainkan menjadi 35 karakter?
7. Bagaimana caranya agar semua kota Yogya yang ada pada

tabel tblpengarang diubahmenjadi Yogyakarta?

8. Apa perintah untuk menghapus data pegawai pada tabel tblpegawai yang NIP-nya adalah123456?

Diinginkan untuk menghapus tabel tblpegawai. Apa perintahnya?

BAB 8
QUERY

Pokok Bahasan
- Sintax SUbquery
- Menggunakan Subquery
- Pedoman SUbquery
- Operator ANY
- Operator ALL

8.1. PENDAHULUAN

Suatu *subquery* adalah suatu pernyataan SELECT yang dilekatkan di dalam suatu klausa pada pernyataan SELECT lain. *Subquery* bisa sangat bermanfaat ketika diperlukan untuk memilih baris-baris dari suatu tabel dengan suatu kondisi yang tergantung pada data di dalam tabel itu sendiri. Kita dapat menempatkan *subquery* di dalam sejumlah klausa klausa SQL, termasuk berikut:

1. Klausa *WHERE*
2. Klausa *HAVING*
3. Klausa *FROM*

Di dalam Syntax: *Operator* termasuk suatu kondisi pembanding seperti >, =, atau IN

Kondisi kondisi pembanding dibagi dalam dua kelas: *singlerow Operator* (>,=,>=,<,<>,<=) dan m*ultiplerow operator* (IN, ANY, ALL).

Subquery lebih dikenal sebagai suatu *SELECT* bersarang (*nested*), *subSELECT*, atau pernyataan *inner* SELECT. Secara umum s*ubquery* dieksekusi pertama kali, dan hasilnya digunakan untuk melengkapi kondisi query pada *query* utama (atau *outer*).

SELECT select_listFROM table
WHERE expr_operator(*SELECT select_list FROM table*)

Subquery (*inner query*) dieksekusi sekali sebelum query utama

(*outer query*). Hasil dari *subquery* digunakan oleh *query* utama.

8.2 TUJUAN PEMBELAJARAN

Setelah mempelajari bab ini, Anda diharapkan mampu:

1. CPMK

Melakukan pemodelan data dan mendemontrasikan penggunaan SQL secara interaktif untuk melakukan berbagai operasi terhadap Basis Data

2. SUB CPMK

Melakukan perumusan dan mendemontrasikan fungsi dalam suatu *syntax* dan fungsi SQL serta manipulasi data serta perintah yang digunakan untuk pendefinisian objek Basis Data (DDL)

8.3. SINTAK SUBQUERY

Contoh Kasus:

Siapakah yang memiliki penghasilan lebih dari Abel?
Jawabannya:

Query Utama

Pegawai mana yang memiliki penghasilan lebih dari Abel?

Sub Query

Berapakah penghasilan Abel?Tipe Tipe dari *Subquery*

1. *Singlerow subquery*
   Query yang mengembalikan hanya satu baris dari pernyataan *inner SELECT* (*SELECT* terdalam).
2. *Multiplerow subquery*
   Query yang mengembalikan lebih dari satu baris dari pernyataan *inner* SELECT.

8.4. MENGGUNAKAN SUATU *SUBQUERY* UNTUK MEMECAHKAN SUATU PERSOALAN

Misalkan ingin menulis suatu *query* untuk mencari tahu penghasilan siapa yang lebih besar daripada penghasilan Abel.

Untuk memecahkan masalah ini, diperlukan dua *query*: satu *query* untuk mencari berapa banyak penghasilan Abel, dan *query* kedua untuk mencari penghasilan siapa yang lebih besar dari jumlah itu.

Kita dapat memecahkan persoalan ini dengan menggabungkan dua *query*, menempatkan satu query di dalam *query* lain.

Inner query (atau *subquery*) mengembalikan suatu nilai yang digunakan *outer query* (atau *query* utama). Penggunaan suatu *subquery* sama dengan penggunaan dua *query* berturut turut dan menggunakan hasil dari *query* pertama sebagai nilai pencari dalam *query* yang kedua.

SELECT last-nameFROM employees WHERE salary >
(*SELECT salary FROM employees*
WHERE last_name = ‘Abel’);

8.5. PEDOMAN UNTUK MENGGUNAKAN SUBQUERY

1. *Subquery* diapit tanda kurung.

2. Tempatkan *subquery* di sebelah kanan dari kondisi pembanding.

3. Klausa ORDER BY dalam *subquery* tidak diperlukan kecuali dilakukan pemeringkatan (*TopNanalysis*).

4. Gunakan *singlerow* operator pada *singlerow subquery*, dan gunakan *multiple row* operator pada *multiple row subquery.*

5. Single Row Subqueries

Suatu *singlerow subquery* adalah mengembalikan satu baris dari pernyataan *inner* SELECT.

Tipe dari *subquery* ini menggunakan suatu *singlerow operator*.Beberapa operator pembanding *single row:*

operator
=
>
>=
<
<=
<>

Tampilkan pegawai-pegawai yang job Idnya sama dengan pegawai 141: *SELECT last_name, job_id*

FROM employees

WHERE job_id = (*SELECT job_id*

FROM employees

WHERE employee_id = 141);

8.6. OPERATOR ANY

(Dan sinonimnya, operator *SOME*) membandingkan suatu nilai pada *setiap* nilai yang dikembalikan oleh suatu *subquery*. Contoh di bawah menampilkan para pegawai yang bukan IT programmer dan penghasilan siapa yang kurang dari beberapa IT programmer. Penghasilan maksimum yang didapat seorang programmer adalah $ 9,000.

<*ANY* maksudnya kurang dari maksimum. >*ANY* maksudnya lebih dari minimum. =*ANY* adalah sama dengan IN.

SELECT employee_id, last_name, job_id, salary FROM employee

WHERE salary < ANY (*SELECT salary*

FROM employee

WHERE job_id = 'IT PROG') *AND job_id <>*

'IT_PROG'

8.7. OPERATOR ALL

Membandingkan suatu nilai *untuk setiap* nilai yang dikembalikan oleh suatu *subquery*. Contoh menampilkan para pegawai yang penghasilannya kurang dari penghasilan dari semua pegawai dengan suatu job ID IT_PROG dan siapa yang bukan IT_PROG.

>*ALL* maksudnya lebih dari maksimum, dan <*ALL* maksudnya kurang dari minimum. Operator NOT dapat digunakan dengan operator-operator *IN, ANY, dan ALL*. *SELECT employee_id, last_name, job_id, salary*
FROM employee
WHERE salary < ALL (*SELECT salary*
FROM employee
WHERE job_id = 'IT PROG')
AND job_id <> 'IT_PROG'

BAB 9
INTEGRITAS BASIS DATA

Pokok Bahasan
- Integritas Keunikan Data
- Integritas Domain data
- Integritas Aturan Nyata

9.1. PENDAHULUAN

Sebagai sarana untuk meyakinkan bahwa nilai-nilai data dalam sistem Basis Data selalu benar, konsisten, selalu tersedia. *Integrity constraints* (batasan integritas) menjaga dari kejadian yang merusak *database*, dengan menjamin perubahan *database* tidak menghasilkan kehilangan konsistensi data. Domain *constraints* adalah bentuk paling umum dari *integrity constraint.* Dapat dilakukan dengan cara:

1. Pastikan bahwa nilai-nilai data adalah benar sejak dimasukkan pertama kali
2. Membuat program untuk mengecek keabsahan data pada saat dimasukkan kecompute
3. Penolakan/pembatalan aksi (*cancelation*)
4. Pengisian nilai kosong pada *field* tertentu (*nullify*)
5. Penjalaran perubahan (*cascade*)

9.2. TUJUAN PEMBELAJARAN

Setelah mempelajari bab ini, Anda diharapkan mampu:

1. CPMK

Mengidentifikasi nilai-nilai islam kedalam pengelolaan data dalam menjaga keamanan data dan integritas suatu data agar tidak menjadi kebocoran data dalam penerapan suatu program bisnis

2. SUB CPMK

Mampu menjelaskan hadis dan ayat Al-Qur'an yang berhubungan dengan keamanan data dan integritas suatu data agar tidak menjadi kebocoran data dalam penerapan suatu program bisnis

9.3. INTEGRITAS KEUNIKAN DATA

Dilakukan melalui:

a. Pendefinisian struktur tabel dengan membuat indeks primer yang bersifat unik
b. Pengkodean di dalam aplikasi pada saat pemasukan/ penambahan data lebih *user-friendly*
c. Kedua cara diterapkan bersama-sama

9.3 INTEGRITAS DOMAIN DATA

Dilakukan melalui:

1. Penetapan tipe data pada setiap *field* di dalam tabel
2. Pengisian *validation rule* dari DBMS Integritas Referensial/*Referential Integrity* (relasi antar tabel)
3. Memastikan sebuah nilai yang muncul pada suatu tabel dari atribut yg diberikan jugamuncul pada atribut tabel lain.
    a. Harus selalu dijaga karena kesalahan referensial dapat menimbulkan kesalahan barudalam Basis Data

b. Dilakukan pengecekan pada proses penambahan, pengubahan, dan penghapusan data.

Contoh: jika "Perryridge" adalah nama cabang pada satu baris di tabel *account*, maka ada sebuah baris di tabel *branch* yang berisi "Perryridge".

Contoh *Referential Integrity* pada SQL

create table customer (customer-name char(20)**,** customer-street char(30),customer-city char(30), **primary key** (customer-name))

create table branch

(branch-name char(15)**,** branch-city char(30), assets integer,

primary key (branch-name))

Contoh Referential Integrity pada SQL

create table account

(account-number char(10)**,** branch-name char(15), balance integer,

primary key (account-number),

foreign key (branch-name) **references** branch)

create table depositor (customer-name char(20)**,** account-number char(10)**,**

primary key (customer-name, account-number), **foreign key** (account-number) **references** account, **foreign key** (customer-name) **references** customer)

9.5 INTEGRITAS ATURAN NYATA

Sifatnya sangat kasuistik, tidak berlaku umum. Pada kasus yang berbeda, aturannya bisa berbeda pula. Untuk mengakomodasi adanya *business role* ini, dengan menyiapkan tabel khusus yang menampung nilai-nilai konstanta yang dibutuhkan aplikasi pada saat dijalankan yang mudah diubah tanpa mengakibatkan perubahan aplikasi maupun struktur Basis Data

GLOSARIUM

- *Enterprise:* suatu bentuk organisasi
- Entitas: suatu objek yang dapat dibedakan dengan objek lainnya
- Atribute/*field*: setiap entitas mempunyai atribut atau suatu sebutan untuk mewakili suatu entitas.
- *Data value:* data aktual atau informasi yang disimpan pada tiap data elemen atau atribut.
- *Record/tuple;* kumpulan elemen-elemen yang saling berkaitan menginformasikan tentang suatu *entity* secara lengkap.
- *File*: kumpulan *record-record* sejenis yang mempunyai panjang elemen sama, *attribute* yang sama namun berbeda-beda data *value*-nya.
- Kunci elemen data: tanda pengenal yang secara unik mengindentifikasikan entitas dari suatu kumpulan entitas.

DAFTAR PUSTAKA

Davis, G. B. (1993). *Kerangka Dasar Sistem Informasi Manajemen.* Jakarta: Pustaka Binaman Pressindo.

Fathansyah. (2002). *Basis Data.* Bandung: Informatika Bandung.

Fathansyah. (2015). *Basis Data.* Bandung: Informatika Bandung.

Hariyanto, Bambang. (2004). Sistem Manajemen Basis Data, Informatika, Bandung

Jogiyanto, H.M. (2002). Analisis Dan Desain Sistem Informasi, Andi Offset, Yogyakarta

Kadir, Abdul. (2008). *Belajar Database Menggunakan MySQL*. Yogyakarta: Andi Offset.

Martin, J. (1977). *Database Organization.* United States: Prentice Hall PTR.

Mubarak, Z. Y., Sasongko, M. N., & Syafei, H. (2018). *Analisis Usabilitas Sistem Informasi Akademik di Stikes Al-Irsyad Al-Islamiyyah Cilacap*. UNIVERSITAS AMIKOM Yogyakarta.

Mubarak, Z. Y., Sasongko, M. N., & Syafei, H. (2018). Rancang Bangun Sistem Informasi Posyandu Stunting di Kabupaten Cilacap. *Jurnal Sistem Informasi dan Teknologi Informasi.*

Nugroho, A. (2004). *Konsep Pengembangan Sistem Basis Data.* Bandung: Informatika Bandung.

Nugroho, Adi. (2004). Konsep Perancangan Sistem Basis Data, Andi Offset, Yogyakarta.

Dalam era digitalisasi sekarang ini, kebutuhan akan informasi telah meningkat secara signifikan seiring waktu. Konsep data terus berkembang dan bertransformasi dalam dunia bisnis. Basis data memungkinkan perusahaan untuk mengelola sejumlah besar informasi mengenai setiap pelanggan, sehingga perusahaan dapat memahami perilaku, preferensi, dan kebutuhan pelanggan mereka, baik secara individu maupun kelompok tersegmentasi. Melalui wawasan yang diperoleh dengan menganalisis basis data pelanggan ini, perusahaan jasa dapat melayani pelanggan dengan lebih baik dan dengan demikian menciptakan pelanggan yang lebih setia. Selain itu, manajer juga dapat menganalisis *database* pelanggan untuk mendapatkan wawasan tentang pengembangan produk baru dan untuk menciptakan cara yang lebih baik dalam menyampaikan produk yang sudah ada.

Buku ***Konsep dan Implementasi Basis Data*** ini disusun dengan harapan dapat memberikan kemudahan bagi Anda untuk memahami konsep basis data serta cara mengimplementasikannya. Buku ini ditujukan bagi para mahasiswa yang mengambil mata kuliah Basis Data, bagi Anda yang ingin mempelajari dunia teknologi, serta siapa saja yang ingin mempelajari bagaimana merancang basis data.

Materi yang disajikan diberikan dengan jelas dan terperinci yang disertai dengan contoh penerapannya. Dalam setiap bab diberikan rangkuman dan latihan soal sehingga dapat membantu Anda lebih memahami materi. Pembahasan dalam buku ini meliputi:

- Konsep dan Definisi Basis Data
- Kelebihan dan Kriteria Basis Data
- Implementasi Basis Data dalam Program/Aplikasi
- Sistem Basis Data (*Database Management System*)
- Pemodelan Data ERD, UML dan DDL
- Keamanan dan Integritas Data

UNAIC PRESS
CILACAP

UNAIC PRESS CILACAP
Jl. Cerme No.24, Wanasari
Kabupaten Cilacap, Jawa Tengah 53223
Telp/Fax: (0282) 532975
Email: humas@universitasalirsyad.ac.id

ISBN 978-623-88026-1-6