BAB I PENGERTIAN

- BASIS DATA ATAU DATABASE BERASAL DARI KATA :

  BASIS DAN DATA

- BASIS → MARKAS ATAU GUDANG, TEMPAT BERSARANG ATAU BERKUMPUL, DASAR
- DATA → REPRESENTASI FAKTA DUNIA NYATA SUATU OBJEK SEPERTI MANUSIA (PEGAWAI, SISWA, PEMBELI, PELANGGAN), BARANG, HEWAN DLL DIREKAM DALAM BENTUK ANGKA, HURUF, SIMBOL, TEKS, GAMBAR, BUNYI ATAU KOMBINASI

PENGERTIAN BASIS DATA

- BASIS DATA ADALAH KUMPULAN FILE / TABEL YANG SALING BERINTERAKSI DAN DAPAT DIGUNAKAN BERSAMA.

- TUJUAN DIBENTUKNYA BASIS DATA ADALAH KEMUDAHAN DAN KECEPATAN DALAM PENGAMBILAN KEMBALI DATA (RETRIEVAL).

- SUATU MEDIA PENYIMPANAN (HARD DISK) DAPAT MENEMPATKAN LEBIH DARI 1 (SATU) BASIS DATA DAN TIDAK SEMUA BENTUK PENYIMPANAN DATA SECARA ELEKTRONIK DIKATAKAN BASIS DATA

HIRARKI DATA

- DIKELOMPOKKAN MENJADI 3 (TIGA) BUAH :
  1. FILE/TABEL/BERKAS.
  2. RECORD/REKAMAN/BARIS.
  3. ELEMEN DATA/FIELD/ATRIBUT

- FILE /TABEL ADALAH KUMPULAN RECORD SEJENIS YANG MEMPUNYAI PANJANG ATRIBUT / FIELD SAMA, NAMUN BERBEDA ISI DATANYA.

HIRARKI DATA

- RECORD ADALAH SEKUMPULAN ELEMEN DATA/FIELD YANG SALING TERKAIT

  CONTOH : NIS, NAMA, TGL_LHR, ALAMAT DAN ATRIBUT LAINNYA DARI SISWA DAPAT DIHIMPUN DALAM SEBUAH RECORD / BARIS.
- FIELD/ ATRIBUT ADALAH SATUAN DATA TERKECIL YANG TIDAK DAPAT DIPECAH LAGI MENJADI UNIT LAIN YANG BERMAKNA

SISTEM BASIS DATA

- SEKUMPULAN FILE/TABEL YANG SALING BERHUBUNGAN.
- KOMPONEN-KOMPONENNYA :

1. HARDWARE.
2. OPERATING SYSTEM.
3. DATABASE.
4. SISTEM (APLIKASI/PERANGKAT LUNAK) PENGELOLA BASIS DATA (DBMS)
5. USER.
6. APLIKASI (PERANGKAT LUNAK) LAIN (BERSIFAT OPTIONAL)

SISTEM PENGELOLA BASIS DATA (DBMS)

- PENGELOLAAN BASIS DATA SECARA FISIK TIDAK DITANGANI LANGSUNG OLEH USER, TETAPI DITANGANI OLEH PERANGKAT LUNAK (SISTEM) YANG KHUSUS/SPESIFIK DISEBUT DBMS.
- DBMS MENENTUKAN BAGAIMANA DATA DIORGANISASI, DISIMPAN, DIUBAH DAN DIAMBIL KEMBALI.
- DBMS ADALAH KOLEKSI TERPADU DARI PROGRAM-PROGRAM (SISTEM PERANGKAT LUNAK) YANG DIGUNAKAN UNTUK MENDEFINISIKAN, MENCIPTAKAN, MENGAKSES DAN MERAWAT DATABASE
- CONTOH DBMS ADALAH MYSQL, MariaDB, Ms ACCESS, Ms SQL SERVER DAN ORACLE.

BAB 2 OPERASI DASAR

- CREATE DATABASE
- DROP DATABASE
- CREATE TABLE
- DROP TABLE
- INSERT
- RETRIEVE/SEARCH
- UPDATE
- DELETE

KEGUNAAN DATABASE

1. REDUNDANSI DAN INKONSISTENSI DATA
2. KESULITAN PENGAKSESAN DATA
3. ISOLASI DATA UNTUK STANDARISASI.
4. BANYAK PEMAKAI (MULTIPLE USER)
5. MASALAH KEAMANAN (SECURITY)
6. MASALAH INTEGRASI (KESATUAN)
7. MASALAH DATA INDEPENDENCE (KEBEBASAN DATA)

REDUNDANSI DAN INKONSISTENSI DATA

- BEBERAPA BAGIAN DATA MENGALAMI PENGGANDAAN PADA TABEL YANG BERBEDA DI DATABASE.
- PENYIMPANAN DATA YANG SAMA (BERULANG-ULANG) DI BEBERAPA TEMPAT DALAM DATABASE DAPAT MENGAKIBATKAN INKONSISTENSI DATA (TIDAK KONSISTEN DATA).

HIRARKI DATA

- RECORD ADALAH SEKUMPULAN ELEMEN DATA/FIELD YANG SALING TERKAIT

  CONTOH : NIM, NAMA, TGL_LHR, ALAMAT DAN ATRIBUT LAINNYA DARI MAHASISWA DAPAT DIHIMPUN DALAM SEBUAH RECORD / BARIS.
- FIELD/ ATRIBUT ADALAH SATUAN DATA TERKECIL YANG TIDAK DAPAT DIPECAH LAGI MENJADI UNIT LAIN YANG BERMAKNA

KESULITAN PENGAKSESAN DATA

- KESULITAN AKAN TIMBUL PADA SAAT BELUM TERSEDIA PROGRAM, SEHINGGA PENYELESAIAN OLEH DBMS YANG MAMPU MENGAMBIL DATA SECARA LANGSUNG DENGAN BAHASA YANG FAMILIAR DAN MUDAH DIGUNAKAN (USER FRIENDLY)

ISOLASI DATA UNTUK STANDARISASI

- DATA DALAM FILE/TABEL PADA BENTUK FORMAT YANG TIDAK SAMA, MAKA SULIT DALAM MENULIS PROGRAM APLIKASI UNTUK MENGAMBIL DAN MENYIMPAN DATA, MAKA HARUSLAH DATA DALAM SATU BASIS DATA DIBUAT SATU FORMAT SEHINGGA MUDAH DIBUAT PROGRAM APLIKASINYA.

MULTIPLE USER

DATA YANG DIGUNAKAN BERSAMA DALAM WAKTU YANG SAMA ATAU BERBEDA DAN DIAKSES OLEH PROGRAM YANG SAMA TAPI BERBEDA ORANG DAN WAKTU

MASALAH INTEGRITAS (KESATUAN)

- BASIS DATA BERISI FILE / TABEL YANG SALING TERKAIT, SECARA TEKNIS FIELD / ATRIBUT KUNCI YANG MENGAITKAN / MERELASIKAN TABEL TERSEBUT.

DATA INDEPENDENCE (KEBEBASAN DATA)

- MELAKUKAN PERUBAHAN PADA STRUKTUR FILE/ TABEL, LIHAT DATA DENGAN UTILITY LIST, MENAMBAH DATA DENGAN APPEND.
- PERINTAH-PERINTAH DALAM PAKET DBMS BEBAS TERHADAP BASIS DATA.
- PERUBAHAN DALAM BASIS DATA SEMUA PERINTAH AKAN MENGALAMI KESTABILAN TANPA MENGALAMI PERUBAHAN.

KEUNTUNGAN SISTEM BASIS DATA

- MENGURANGI REDUDANSI DATA → KERANGKAPAN DATA.
- INTEGRITAS DATA → AKURAT DATA.
- MENGHINDARI INKONSISTEN DATA.
- DATA SHARE → DATA DIGUNAKAN BERSAMA.
- STANDARISASI DATA.
- JAMINAN SECURITY DATA
- MENYEIMBANGKAN KEBUTUHAN DATA.

KERUGIAN SISTEM BASIS DATA

- DIPERLUKAN TAMBAHAN HARDWARE → CPU YANG LEBIH KUAT, TERMINAL YANG LEBIH BANYAK, ALAT KOMUNIKASI.
- BIAYA PERFORMANCE YANG LEBIH BESAR → LISTRIK, KARYAWAN YANG LEBIH TINGGI KLASIFIKASINYA, BIAYA TELEKOMUNIKASI ANTAR LOKASI AKAN BERTAMBAH
- RAWANNYA KEBERHASILAN OPERASI → GANGGUAN LISTRIK DAN KOMUNIKASI.
- SISTEM MENJADI LEBIH KOMPLEKS → BANYAKNYA ASPEK YANG HARUS DIPERHATIKAN.

MANIPULASI DATA

- INSERT
- DELETE
- UPDATE
- RETRIEVE

ABSTRAKSI DATA

- KEGUNAAN : AGAR PEMAKAI (USER) MAMPU MENYUSUN PANDANGAN ABSTRAKSI DARI DATA.
- DIKELOMPOK MENJADI TIGA TINGKATAN YAITU
- LEVEL PHISIK → MENGGAMBARKAN DATA DISIMPAN DALAM KONDISI SEBENARNYA.
- LEVEL KONSEPTUAL → MENGGAMBARKAN DATA APA YANG DISIMPAN DALAM BASIS DATA DAN HUBUNGAN RELASI YANG TERJADI ANTAR DATA.

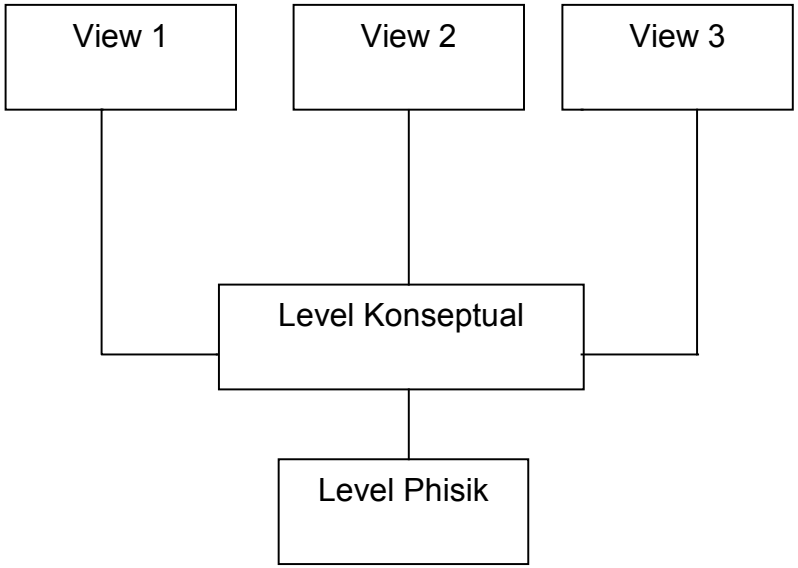
ABSTRAKSI DATA

3. LEVEL PANDANGAN PEMAKAI (VIEW LEVEL).

- LEVEL ABSTRAKSI TERTINGGI YANG MENGGAMBARKAN HANYA SATU BAGIAN DARI KESELURUHAN.
- LEVEL INI SANGAT DEKAT DENGAN PEMAKAI (USER) DAN SETIAP USER KEMUNGKINAN HANYA MEMBUTUHKAN SEBAGIAN DARI DATABASE.
- MISALKAN PEMAKAI AKHIR PADA BAGIAN KEUANGAN HANYA MEMAKAI DATA UNTUK FILE/TABEL PEMBAYARAN, MAHASISWA DAN KARYAWAN.
- BEBERAPA PANDANGAN DISUSUN UNTUK MENGAKSES SATU SISTEM DATABASE YANG SAMA.

ABSTRAKSI DATA

Bag Keuangan Bag Perpustakaan Bag Akademik



BAB III.
DATABASE LANGUAGE

KOMPONEN BAHASA BASIS DATA

1. DATA DEFINITION LANGUAGE (DDL).
2. DATA MANIPULATION LANGUAGE (DML)
3. DATA CONTROL LANGUAGE (DCL).

DATA DEFINITION LANGUAGE

- STRUKTUR / SKEMA BASIS DATA YANG MENGGAMBARKAN/MEWAKILI DESAIN BASIS DATA SECARA KESELURUHAN DISPESIFIKASI DENGAN BAHASA KHUSUS.
- DENGAN BAHASA INI DAPAT MEMBUAT TABEL (CREATE TABLE), INDEKS MENGUBAH TABEL
- MENENTUKAN STRUKTUR PENYIMPANAN TABEL DAN LAINNYA.
- KAMUS DATA ADALAH KUMPULAN TABLE YANG DISIMPAN DALAM FILE KHUSUS.
- CONTOH PERINTAH DDL DENGAN FOXPRO ADALAH :
  * CREATE
  * MODIFY REPORT
  * MODIFY STRUCTURE.

DATA MANIPULATION LANGUAGE

- UNTUK MELAKUKAN MANIPULASI DAN PENGAMBILAN DATA PADA SUATU BASIS DATA BERUPA :

1. INSERT → PENYISIPAN/PENAMBAHAN DATA
2. DELETE → HAPUS DATA.
3. UPDATE → UBAH DATA.
4. SEARCH → PENCARIAN / PENELUSURAN DATA

JENIS-JENIS DML

1. PROSEDURAL

- MENSYARATKAN PEMAKAI MENENTUKAN, DATA APA YANG DIINGINKAN SERTA BAGAIMANA CARA MENDAPATKANNYA.

2. NON PROSEDURAL

- MEMBUAT PEMAKAI DAPAT MENENTUKAN DATA APA YANG DIINGINKAN TANPA MENYEBUTKAN BAGAIMANA CARA MENDAPATKANNYA.

CONTOH PAKET BHS PROSEDURAL DML : Dbase, FoxBase.

Non Prosedural DML : SQL, QBE.

QUERY

- PERNYATAAN YANG DIAJUKAN UNTUK MENGAMBIL INFORMASI DI DALAM SUATU BASIS DATA.
- MERUPAKAN BAGIAN DARI DML UNTUK PENGAMBILAN KEPUTUSAN.
- CONTOH PENGGUNAAN PERINTAH QUERY (SQL).

```
SELECT
  NID,NAMA_D,JKELAMIN,ALAMAT,KOTA
FROM DOSEN
WHERE JKELAMIN='PRIA'
```

PENGGUNA DATABASE

1. DATABASE MANAGER.

   SATU DATABASE MANAGER ADALAH SATU MODUL PROGRAM YANG MENYEDIAKAN INTERFACE ANTARA PENYIMPANAN DATA LOW-LEVEL DALAM DATABASE DENGAN SATU APLIKASI PROGRAM DAN QUERY YANG DIAJUKAN KE SISTEM.

   TUGAS DAN TANGGUNGJAWAB YAITU :

   a. INTERAKSI DENGAN MANAGER FILE.

   b. INTEGRITAS.

   c. KEAMANAN.

   d. BACKUP DAN RECOVERY.

PENGGUNA DATABASE

2. DATABASE ADMINISTRATOR.

- PENGONTROLAN TERHADAP SELURUH SISTEM BAIK DATA MAUPUN PROGRAM YANG MENGAKSES DATA.
- FUNGSI DATABASE ADMINISTRATOR (DBA) :

a. MENDEFINISIKAN POLA STRUKTUR DATABASE.
b. MENDEFINISIKAN STRUKTUR PENYIMPANAN DAN METODE AKSES.
c. MAMPU MEMODIFIKASI POLA DAN ORGANISASI PHISIK.
d. MEMBERIKAN KEKUASAAN PADA USER UNTUK MENGAKSES DATA.
e. MENSPESIFIKASIKAN KEHARUSAN INTEGRITAS DATA.

PENGGUNA DATABASE

3. DATABASE USER.
   a. PROGRAMMER APLIKASI.
   b. CASUAL USER (USER MAHIR)
   c. USER UMUM (END USER)
   d. USER KHUSUS (SPECIALIZED USER)

PENETAPAN STRUKTUR TABEL

- NAMA KOLOM (FIELD/ATRIBUT)
- TIPE DATA (DATA TYPE)
- LEBAR BANYAKNYA KARAKTER/DIGIT MAKSIMUM YANG DAPAT DITAMPUNG)
- PENDEFINISIAN KOLOM (APAKAH NULL ATAU NOT NULL)

NORMALISASI DATA

- PROSES NORMALISASI MERUPAKAN PROSES PENGELOMPOKKAN DATA ELEMEN MENJADI TABEL-TABEL YANG MENUNJUKKAN ENTITY DAN RELASINYA.
- BILA ADA KESULITAN PADA PENGUJIAN MAKA RELASI TERSEBUT DIPECAHKAN MENJADI BEBERAPA TABEL LAGI, SEHINGGA DIPEROLEH DATABASE YANG OPTIMAL.

ATRIBUT TABEL

- DIFOKUSKAN PADA TINJAUAN KOMPREHENSIF TERHADAP SETIAP KELOMPOK DATA (TABEL) SECARA INDIVIDUAL
- SUATU FIELD / ATRIBUT DIJADIKAN KEY, MAKA TIDAK BOLEH ADA DUA ATAU LEBIH BARIS DATA DENGAN NILAI YANG SAMA UNTUK FIELD / ATRIBUT TERSEBUT.
- JENI-JENIS KEY YAITU :

1. PRIMARY KEY (PK)
2. ALTERNATE KEY
3. SECONDARY KEY
4. CANDIDATE KEY
5. COMPOSITE KEY
6. FOREIGN KEY

ATRIBUT SEDERHANA

- ATRIBUT ATOMIK YANG TIDAK DAPAT DIPILAH LAGI MENJADI ATRIBUT LAINNYA.
- CONTOH : NIM DAN NAMA PADA TABEL MAHASISWA.

HIRARKI DATA

- RECORD ADALAH SEKUMPULAN ELEMEN DATA/FIELD YANG SALING TERKAIT

  CONTOH : NIM, NAMA, TGL_LHR, ALAMAT DAN ATRIBUT LAINNYA DARI MAHASISWA DAPAT DIHIMPUN DALAM SEBUAH RECORD / BARIS.
- FIELD/ ATRIBUT ADALAH SATUAN DATA TERKECIL YANG TIDAK DAPAT DIPECAH LAGI MENJADI UNIT LAIN YANG BERMAKNA

ATRIBUT KOMPOSIT

- ATRIBUT YANG MASIH DAPAT DIUARIKAN LAGI MENJADI SUB-SUB ATRIBUT YANG MASING-MASING MEMILIKI MAKNA.
- CONTOH : ALAMAT PADA TABEL MAHASISWA MASIH BISA DIURAIKAN MENJADI BEBERAPA SUB ATRIBUT SEPERTI : ALMAT_JL,KELURAHAN,KECAMATAN,Rt,Rw, NO_RUMAH YANG MASING-MASING MEMILIKI MAKNA TERSENDIRI.

ATRIBUT BERNILAI TUNGGAL

- DITUJUKAN PADA ATRIBUT-ATRIBUT YANG MEMILIKI PALING BANYAK SATU NILAI UNTUK SETIAP BARIS DATA.
- ATRIBUT YANG HANYA DAPAT BERISI SATU NILAI.

ATRIBUT BERNILAI BANYAK

- DITUJUKAN PADA ATRIBUT-ATRIBUT YANG DAPAT DIISI LEBIH DARI SATU NILAI, TETAPI JENISNYA SAMA.

ATRIBUT HARUS BERNILAI (MANDATORY ATTRIBUTE)

- MERUPAKAN SEJUMLAH ATRIBUT YANG ADA PADA SUATU TABEL YANG HARUS BERISI DATA DAN TIDAK BOLEH KOSONG.
- NON MANDATORY ATTRIBUTE ADALAH SEJUMLAH ATRIBUT YANG ADA PADA SUATU TABEL YANG BOLEH TIDAK DIISI DATANYA/BOLEH KOSONG.
- NILAI NULL DIGUNAKAN UNTUK MENGISI ATRIBUT-ATRIBUT YANG NILAINYA MEMANG BELUM SIAP/TIDAK ADA.

ATRIBUT TURUNAN (DERIVED ATTRIBUTE)

- ATRIBUT YANG NILAI-NILAINYA DIPEROLEH DARI PENGOLAHAN ATAU DAPAT DITURUNKAN DARI ATRIBUT ATAU TABEL LAIN YANG BERHUBUNGAN.

ORGANISASI FILE

- FILE DIORGANISASI (DISUSUN) BERDASARKAN URUTAN-URUTAN RECORD-RECORD.
- RECORD-RECORD DIPETAKAN KE DALAM BLOK-BLOK DALAM HARDDISK
- BLOK BERUKURAN TETAP, 1 BLOK BERISI LEBIH DARI 1 RECORD
- JENIS RECORD BERDASARKAN PANJANGNYA :
  - FIXED LENGTH RECORD
  - VARIABLE LENGTH RECORD

RECORD 1	0411500005	Ahmad Zaki	Cipondoh
RECORD 2	0422500025	Sinta	Kebayoran Lama
RECORD 3	0422500035	Indra Gunawan	Cipulir
RECORD 4	0433500058	Bekti Sularso	Cidodol
RECORD 5	0444500057	Tini Lestari	Cileduk

FIXED LENGTH RECORD

- RECORD YANG PANJANGNYA TETAP
- MISAL : UNTUK MEMBUAT RECORD MAHASISWA

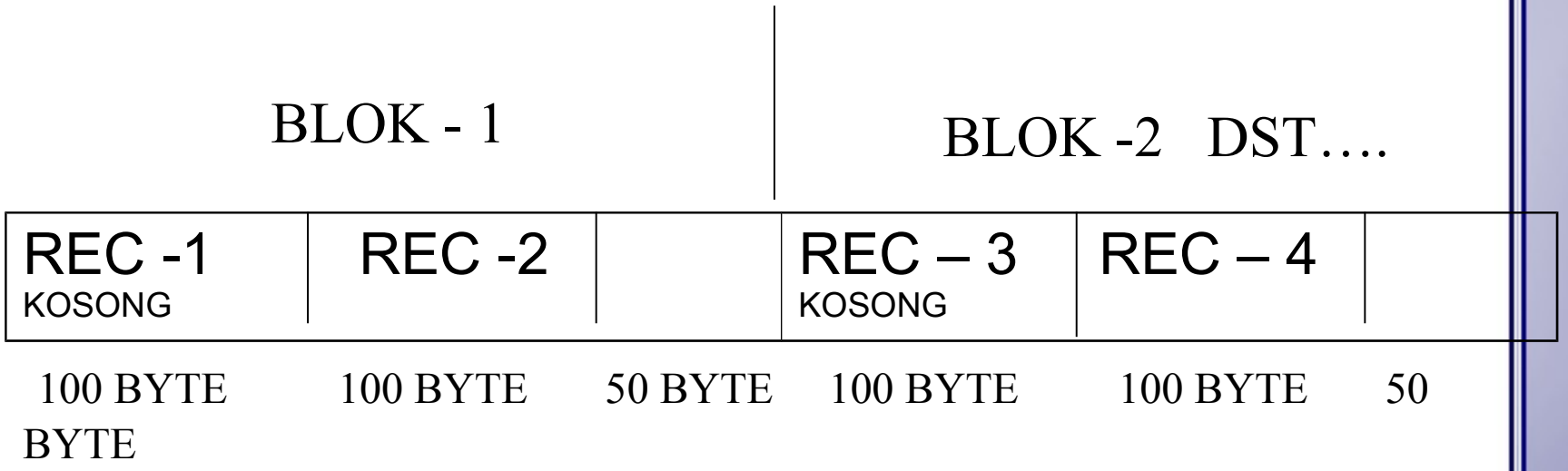
```
TYPE MAHASISWA = RECORD
                    NIM : CHAR(10);
                    NAMA : CHAR(40);
                    ALAMAT : CHAR(50);
                END
```

- TIAP KARAKTER MENYIMPAN 1 BYTE, MAKA RECORD KE 1 UNTUK DATA MAHASISWA DI ATAS AKAN MENYIMPAN 100 BYTE, KEMUDIAN 100 BYTE UNTUK RECORD YANG KEDUA DAN SETERUSNYA.

FIXED LENGTH RECORD

- PENEMPATAN RECORD PADA BLOK DISEBUT BLOCKING
- METODE BLOCKING UNTUK RECORD BERUKURAN TETAP ADALAH FIXED LENGTH BLOCKING
- MISAL :

  1 BLOCK DAPAT MENYIMPAN 250 BYTE, JIKA 1 RECORD PANJANGNYA 100 BYTE MAKA BLOCKING SBB:



FIXED LENGTH RECORD

- KELEBIHAN FIXED LENGTH RECORD :

  MUDAH DALAM PEMROGRAMAN, KARENA UNTUK MENYISIPKAN ATAU MENGHAPUS RECORD MUDAH KARENA PANJANG RECORDNYA SAMA

- KEKURANGAN FIXED LENGTH RECORD :

  BOROS TEMPAT PENYIMPANAN

VARIABLE LENGTH RECORD

- RECORD YANG PANJANGNYA TIDAK TETAP
- MISAL : UNTUK MEMBUAT RECORD MAHASISWA

```
TYPE MAHASISWA = RECORD
                     NIM : CHAR(10);
                     NAMA : CHAR(40);
                     ALAMAT : CHAR(50);
                END
```

- PANJANG TIAP RECORD BERBEDA-BEDA TERGANTUNG DARI ISI DARI MASING-MASING RECORD
- PENEMPATAN RECORD DALAM BLOK TERGANTUNG DARI PANJANG RECORD
- METODE BLOCKING UNTUK RECORD BERUKURAN TIDAK TETAP ADA DUA :
  - VARIABLE LENGTH SPANNED BLOCKING
  - VARIABLE LENGTH UNSPANNED BLOCKING

VARIABLE LENGTH RECORD

RECORD 1	0411500005	Ahmad Zaki	Cipondoh
RECORD 2	0422500025	Sinta	Kebayoran Lama
RECORD 3	0422500035	Indra Gunawan	Cipulir
RECORD 4	0433500058	Bekti Sularso	Cidodol
RECORD 5	0444500057	Tini Lestari	Cileduk

PANJANG RECORD 1 = 28 BYTE

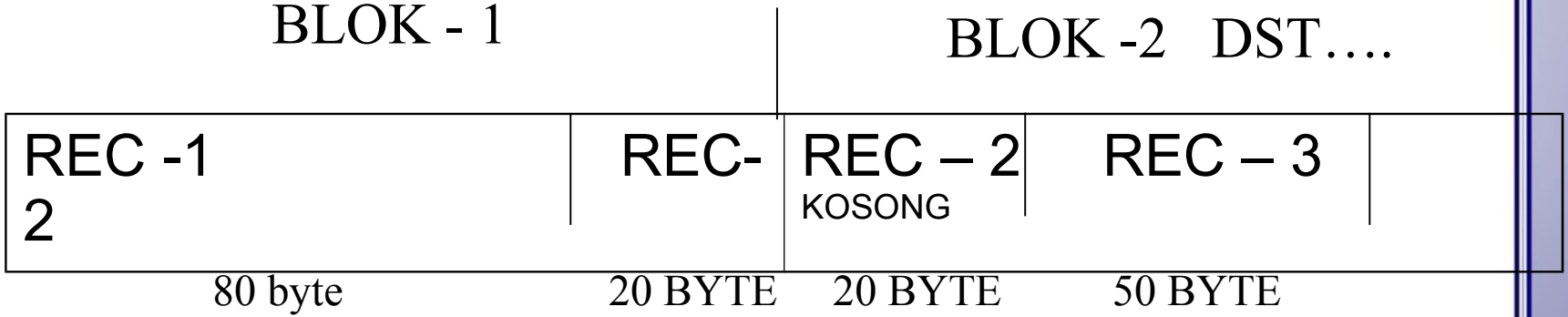
PANJANG RECORD 2 = 29 BYTE

PANJANG RECORD 3 = 30 BYTE DST…

VARIABLE LENGTH RECORD

- VARIABLE LENGTH SPANNED BLOCKING :
  RECORD DITEMPATKAN DALAM BLOK SESUAI DENGAN UKURANNYA JIKA PANJANG RECORD TIDAK DAPAT DIMUAT DALAM 1 BLOK MAKA RECORD DAPAT MUAT DALAM BLOK TERPISAH (1 RECORD DAPAT DIPOTONG)

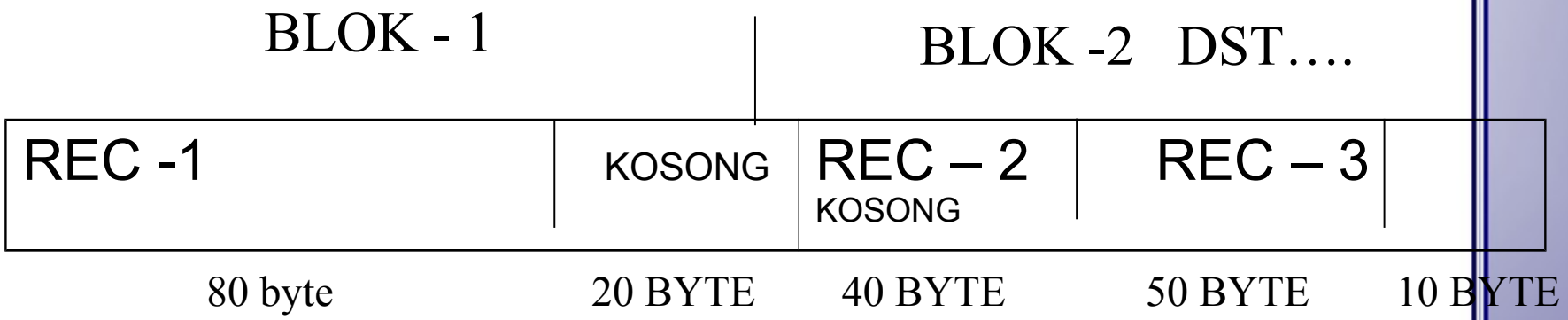
- MISAL : 1 BLOK DAPAT MEMUAT 100 BYTE.
  - PANJANG RECORD 1 = 80 BYTE
  - PANJANG RECORD 2 = 40 BYTE
  - PANJANG RECORD 3 = 50 BYTE



VARIABLE LENGTH RECORD

- VARIABLE LENGTH UNSPANNED BLOCKING :
  RECORD DITEMPATKAN DALAM BLOK SESUAI DENGAN UKURANNYA JIKA PANJANG RECORD TIDAK DAPAT DIMUAT DALAM 1 BLOK MAKA RECORD DAPAT MUAT DALAM BLOK TERPISAH (1 RECORD TIDAK BOLEH DIPOTONG)

- MISAL : 1 BLOK DAPAT MEMUAT 100 BYTE.
  - PANJANG RECORD 1 = 80 BYTE
  - PANJANG RECORD 2 = 40 BYTE
  - PANJANG RECORD 3 = 50 BYTE



VARIABLE LENGTH RECORD

- KELEBIHAN VARIABLE LENGTH RECORD :

  HEMAT TEMPAT PENYIMPANAN

- KEKURANGAN VARIABLE LENGTH RECORD :

  SULIT DIGUNAKAN DALAM PEMROGRAMAN, KARENA PANJANG RECORD BERBEDA MAKA TIAP AKHIR RECORD DIGUNAKAN SYMBOL END OF RECORD YANG MENANDAKAN RECORD SUDAH BERAKHIR

BAB 6

ORGANISASI RECORD-RECORD DALAM FILE

- RECORD TERSUSUN DALAM SEBUAH FILE
- BEBERAPA CARA PENGORGANISASIAN (PENYUSUNAN) RECORD DALAM SEBUAH FILE ADALAH SEBAGAI BERIKUT :
  - ORGANISASI FILE HEAP
    - TIAP RECORD DITEMPATKAN DI MANA SAJA DI DALAM FILE SELAMA MASIH TERDAPAT TEMPAT UNTUK RECORD TERSEBUT
    - TIDAK ADA PENGURUTAN DALAM RECORD
  - ORGANISASI FILE SEKUENTIAL
    - PENEMPATAN RECORD DIURUTKAN SEKUENTIAL BERDASARKAN SEBUAH KEY
  - ORGANISASI FILE HASHING
    - FUNGSI HASH YANG MENGHITUNG BEBERAPA ATTRIBUT DARI RECORD. HASIL DARI FUNGSI AKAN MENEMPATKAN LOKASI DARI RECORD TERSEBUT

ORGANISASI RECORD-RECORD DALAM FILE

BEBERAPA KONSEP DASAR

- FIELD

  SATUAN INFORMASI TERKECIL YANG MENYUSUN RECORD

- RECORD

  KUMPULAN DARI FIELD YANG BERHUBUNGAN SATU SAMA LAIN

- FILE

  KUMPULAN DARI RECORD-RECORD

- BASIS DATA

  KUMPULAN FILE YANG DIGUNAKAN OLEH PROGRAM APLIKASI SERTA MEMBENTUK HUBUNGAN TERTENTU DI ANTARA RECORD-RECORD DI FILE-FILE TERSEBUT

ORGANISASI RECORD-RECORD DALAM FILE

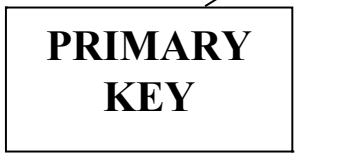
- KEY

  ELEMEN RECORD YANG DIPAKAI UNTUK MENEMUKAN RECORD TERSEBUT PADA WAKTU AKSES

JENIS-JENIS KEY:

PRIMARY KEY

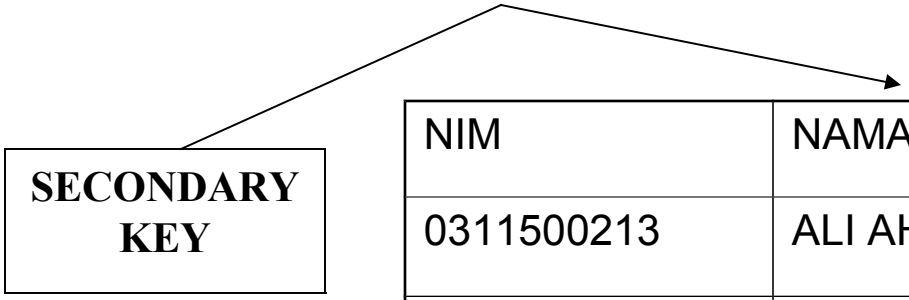
- FIELD YANG MENGIDENTIFIKASIKAN SEBUAH RECORD DALAM FILE
- BERSIFAT UNIK



NIM	NAMA	UMUR
0311500213	ALI AHMAD	19
0322500212	FATIMAH	20

ORGANISASI RECORD-RECORD DALAM FILE

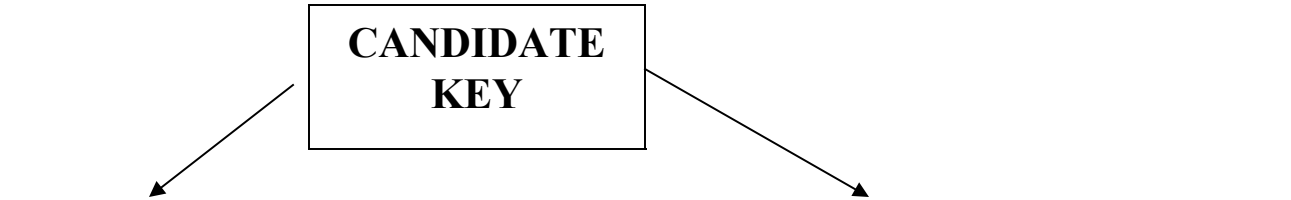
- Secondary key
  - Field yang mengidentifikasikan sebuah record dalam file
  - Tidak bersifat unik



NIM	NAMA	UMUR
0311500213	ALI AHMAD	19
0322500212	FATIMAH	20

ORGANISASI RECORD-RECORD DALAM FILE

- Candidate key
  Field-field yang bisa dipilih (dipakai) menjadi primary key



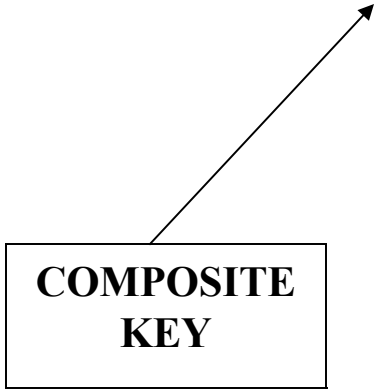
NIM	NAMA	NO. KWITANSI	JUMLAH
0311500012	AHMAD	KW-001	3000000
0322500023	RINA	KW-002	5000000

ORGANISASI RECORD-RECORD DALAM FILE

- Composite key

  Primary key yang dibentuk dari beberapa field

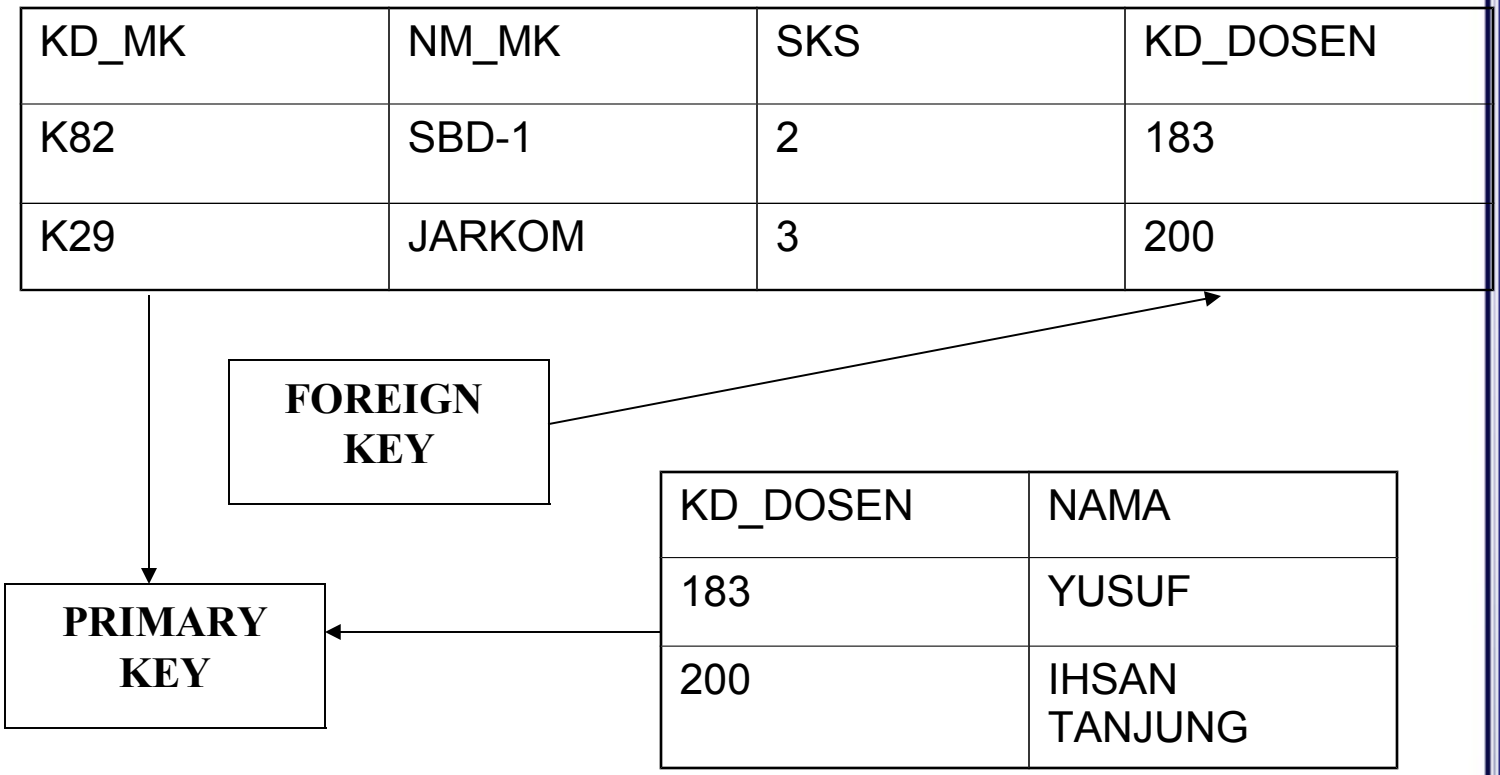
HARI	RUANG	MATA KULIAH
SELASA	4.2.2	JARINGAN KOMPUTER
SELASA	4.2.1	SISTEM BASIS DATA 1
RABU	4.2.2	PANCASILA



ORGANISASI RECORD-RECORD DALAM FILE

- Foreign key

  Field yang bukan key, tetapi adalah key pada file yang lain



ORGANISASI FILE SEKUENTIAL

- FILE SEKUENTIAL DIDESIGN UNTUK EFISIENSI PEMROSESAN REKORD PADA SAAT PENGURUTAN BERDASARKAN BEBERAPA KEY
- FILE DENGAN DATA YANG TERSUSUN DALAM SUATU URUTAN TERTENTU
- TIAP RECORD MEMPUNYAI FIELD YANG SAMA & DENGAN SUSUNAN YANG SAMA

ORGANISASI FILE SEKUENTIAL

STRUKTUR FILE

- UNTUK MEMUNGKINKAN RECORD TERSUSUN SECARA URUT PERLU DITENTUKAN KEY DARI TIAP RECORD

- PEMBACAAN SECARA SERIAL (SATU PERSATU) SESUAI DENGAN URUTAN KEYNYA DISEBUT PEMBACAAN SECARA SEQUENTIAL

Nip	Nama	Pekerjaan
000021	Abu Bakar	Manajer
000032	Fatimah	Sekretaris
000042	Asma	Presiden direktur

ORGANISASI FILE SEKUENTIAL

INSERT SEBUAH RECORD

- INSERT BERARTI MENAMBAHKAN SEBUAH DATA BARU KE DALAM FILE
- INSERT PADA UJUNG AKHIR SEBUAH FILE, HANYALAH MENAMBAH BANYAKNYA DATA WAKTU YANG DIBUTUHKAN KECIL

1	2	3	4	5	6	7	8	9	…
A	B	C	D	E	F	…	…	…	…

INSERT X PADA AKHIR RECORD

1	2	3	4	5	6	7	8	9	…
A	B	C	D	E	F	X	…	…	…

ORGANISASI FILE SEKUENTIAL

- INSERT DITENGAH FILE MENGAKIBATKAN PERGESERAN ATAUP PERUBAHAN STRUKTUR DATA YANG TIDAK SEDERHANA

1	2	3	4	5	6	7	8	9	…
A	B	C	D	E	F	…	…	…	…

INSERT X PADA RECORD KE 3

1	2	3	4	5	6	7	8	9	…
A	B	X	C	D	E	F	…	…	…

→ RECORD KE-3 DST BERGESER

ORGANISASI FILE SEKUENTIAL

DELETE SEBUAH RECORD

- MENGHAPUS SEBUAH RECORD
- MENCARI LOKASI DATA & MENGHAPUS ISINYA, AGAR BISA DIPAKAI OLEH DATA YANG LAIN
- SETELAH ITU DILAKUKAN PERGESERAN ATAUPUN PENGATURAN STRUKTUR DATA KEMBALI

1	2	3	4	5	6	7	8	9	…
A	B	C	D	E	F	…	…	…	…

→ HAPUS

BILA RECORD D DIHAPUS, MAKA AKAN TERJADI PEMBACAAN DAN PENULISAN ULANG RECORD E, F, DST

1	2	3	4	5	6	7	8	9	…
A	B	C	E	F	…	…	…	…	…

ORGANISASI FILE SEKUENTIAL

KADANGKALA DELETE DILAKUKAN DENGAN HANYA MEMBERI TANDA SAJA (TOMBSTONE / FLAG), TANPA DILAKUKAN PENGHAPUSAN ATAUPUN PENGATURAN STRUKTUR DATANYA

1	2	3	4	5	6	7	8	9	…
A	B	C	D	E	F	…	…	…	…

→ **HAPUS**

1	2	3	4	5	6	7	8	9	…
A	B	C	*	E	F	…	…	…	…

→ **record yang sudah dihapus “Delete”**

ANOMALI DAN INTEGRASI DATA PADA MODEL RELASIONAL

Pengertian Anomali :

Proses pada basis data yang berakibat timbul efek samping yang tidak diharap kan.

Contoh efek samping adalah sebagai berikut :

1. **Ketidak-konsistenan data.**
2. **Adanya kehilangan data akibat dari penghapusan data lain.**

Anomali basis data terdiri dari 3 komponen :

1. **Anomali peremajaan (Update).**
2. **Anomali Penghapusan (Delete).**
3. **Anomali penyisipan (Insert).**

Contoh : Relasi KRS (Kartu Rencana Studi) Mahasiswa

Ada 8 atribut : NIM, Nama MHS, Kode Mata Kuliah, Nama Mata Kuliah jumlah SKS, Semester, Tahun, Status ambil.

NIM	Nama MHS	Kode MatKul	Mata Kuliah	SKS	SMT	Tahun	Tahun Ambil
K002	Tukul	mkb023	RPL	8	4	2002	Baru
K003	Peppy	mkb023	RPL	8	4	2002	Ulang
K004	Vega	mkb023	RPL	8	4	2002	Baru
K005	Dian	mkb023	RPL	8	4	2002	Baru
K002	Tukul	ssk027	JarKom	6	4	2002	Baru
K003	Peppy	ssk027	JarKom	6	4	2002	Ulang
K004	Vega	ssk027	JarKom	6	4	2002	Baru
K005	Dian	ssk027	JarKom	6	4	2002	Baru

. Anomali Peremajaan (Update)

Disebabkan oleh perubahan pada sejumlah data yang sia-sia pada sebuah tabel tetapi tidak seluruhnya berubah.

Contoh :
Pada tabel relasi *KRS mahasiswa*, dapat terjadi anomali Update apabila nama mata kuliah *RPL* diubah menjadi *Pancasila*, dan perubahan hanya dilakukan pada record pertama saja dan tidak pada record kedua, ketiga, & keempat, hal ini berakibat adanya ketidakkonsistenan.

. Anomali Penyisipan (insert)

Anomali ini terjadi apabila pada saat penambahan hendak dilakukan, ternyata ada elemen data yang masih kosong dan ternyata elemen data itu adalah *primary key.*

ntoh :

Pada relasi KRS itu terjadi anomali penyisipan, apabila seorang Mahasiswa mengambil beberapa mata kuliah untuk satu semester dan tahun akademik tertentu. Masalahnya bagaimana menyimpan fakta bahwa ada satu mata kuliah baru yang tidak diambil oleh mahasiswa ? Penyisipan tidak bisa dilakukan karena tidak ada informasi mahasiswa yang mengambil mata kuliah tersebut.

Anomali Penghapusan (Delete)

Anomali ini terjadi apabila data pada record ke satu dihapus, maka seluruh data yang ada pada record itu akan terhapus semua, padahal data itu masih dibutuhkan.

Ketergantungan Fungsional (KF)

Diberikan suatu tabel, misal T dengan 2 atribut A dan B, dapat dinyatakan notasi sebagai berikut :

A ➔ B

Pengertian notasi itu adalah : A secara fungsional menentukan B, atau B secara fungsional tergantung pada A.

Diberikan 2 row yaitu : r1 dan r2 dalam tabel T dimana A ➔ B .
Jika r1(A) = r2(A), maka r1(B) = r2(B).

Contoh : Ketergantungan Fungsional (KF)

Row	Nama_Kul	NIM	Nama_MHS	Indeks_Nilai
1	Algoritma	980001	Tono	A
2	Algoritma	980004	Komarudin	B
3	Basis data	980001	Tono	
4	Basis data	980002	Peppy	
5	Basis data	980004	Komarudin	
6	Pengembangan diri	980001	Tono	B
7	Bahasa inggris	980002	Peppy	C

Ketergantungan Fungsional yang dapat terjadi adalah sebagai berikut :

1. **NIM ➔ Nama_MHS, adalah atribut Nama_MHS hanya tergantung pada atribut NIM, faktanya setiap nilai NIM yang sama, maka pasti Nama_MHS juga sama.**

2. Nama_kul, Nim ➔ Indeks_nilai, artinya : atribut indeks_nilai bergantung

Pada atribut Nama_Kul dan Nim bersama – sama.

KF mengandung arti bahwa setiap indeks nilai diperuntukkan pada mahasiswa Tertentu untuk mata kuliah tertentu yang diambilnya.

Contoh Non KF :

1. Nama_Kul ➔ Nim, maksudnya adalah atribut Nim tidak bergantung pada

Nama_Kul. Bukti : pada Row 1 & Row 2, dengan nilai Mata_Kul sama Tetapi nilai Nim nya tidak sama.

2. Nim ↛ Indeks_nilai, artinya adalah bahwa atribut indeks_nilai tidak hanya

bergantung pada atribut Nim. Bukti : pada Row 1 & Row 6, nilai Nim sama Tetapi nilai indeks_nilainya berbeda.

Domain Atribut

Domain atribut adalah suatu gugus nilai yang mungkin dimiliki oleh suatu atribut pada suatu table / relasi dalam database.

Contoh : Relasi rekening pada DataBase perbankan.

	No_Rekening	Status	Saldo
1	012.145.002	Checking	8.000.000
2	012.146.013	Saving	3.000.000
3	012.146.890	Saving	4.000.000
4	210.234.956	Checking	4.860.000

Pada relasi rekening di slide sebelumnya terdapat pengertian sebagai berikut :

1. **Domain dari Status adalah { Saving, Checking }, yang berarti nilai yang mungkin diberikan pada attribut status adalah hanya Saving & Checking.**
2. **Domain dari attribut saldo, adalah semua bilangan nyata yang positif, domain dari Attribut ini adalah gugus tak terhingga.**
3. **Domain attribut No_Rekening adalah semua kode rekening yang mungkin dikeluarkan oleh bank tersebut .**

Integritas Data

Informasi yang disimpan pada basis data hanya bagus bila DBMS membantu pemasukan informasi yang tidak benar.

Konstrain integritas adalah syarat yang dispesifikasikan pada skema basis data & Membatasi data yang disimpan dalam basis data.

Basis data yang memenuhi semua konstrain integritas yang dispesifikasikan pada Skema basis data maka basis data adalah legal.

DBMS memaksakan konstrain integritas sehingga hanya mengijinkan basis data legal yang disimpan oleh DBMS.

Integritas data mengacu pada konsistensi & akurasi data yang di simpan dalam basis data.

Konstrain integritas dapat dibagi 4 bagian , sebagai berikut :

1. **Aturan integritas domain.**
2. **Aturan integritas entitas.**
3. **Aturan integritas referensial.**
4. **Aturan integritas perusahaan (enterprise).**

1. Integritas domain.

Domain adalah nilai – nilai yang dimungkinkan diasosiasikan dengan tiap atribut.
Kemungkinan beberapa atribut memiliki domian yang sama.
Contoh : atribut nama_pelanggan , nama_pekerja dapat mempunyai domain sama. tetapi tidak jelas apakah nama_pekerja & nama_kota seharusnya punya domain yang sama.

2. Integritas Entitas

Null merepresentasikan suatu nilai untuk atribut dimana pada saat itu nilainya, belum diketahui atau tidak diisi. Null tidak sama dengan nilai numerik nol (0) atau String teks spasi.

3. Integritas Referensial

Integritas Referensial adalah jika foreign key terdapat di relasi maka nilai foreign key harus cocok dengan nilai candidate key suatu tupel di relasi asal atau nilai foreign key Seluruhnya Null.

4. Integritas Enterprise

Integritas Enterprise adalah aturan – aturan tambahan yang dispesifikasikan Pemakai atau administrator basis data.

BAB 8

Perancangan Database dengan teknik normalisasi

Bentuk Tak normal

Unnormal Form : Data diambil dari form-form yang ada apa adanya, tidak ada ketentuan mengikuti bentuk tertentu

Contoh Invoice : (terlampir)

Bentuk tak normal

Dirubah ke bentuk Flat :

Invoice No	Nm_Cust	Add_Cust	City_Cust	State_Cust	Zip	Phone_Cust	Date

Order No	Rep	FOB	QTY	Description	Unit Price	Total	Subtotal

Shipping	Tax Rates	G_Total

Bentuk Tak Normal

- Semua atribut/field di susun bentuk flat
- Untuk yang diarsir, memungkinkan adanya data yang bernilai ganda (non atomic value), maka form ini memiliki hubungan *one to many* dengan customer dan barang

Bentuk normal ke satu

- 1st Normal Form : Menghilangkan data bernilai ganda→ menjadi atomic value (bila ada)
- Bentuk ini di uji dengan memasukkan data ke bentuk tak normal, minimal 2 data

Bentuk normal ke satu

Invoice No	Nm_Cust	Add_Cust	City_Cust	State_Cust	Zip	Phone_Cust	Date
0001	ABC	fff	ggg	hhhh	111	123123	1
0002	PQR	Ddd	Eee	Ee	111	1121134	1

Order No	Rep	FOB	QTY	Description	Unit Price	Total	Subtotal
1231	31	3	2	aaaa	30	60	
			3	bbbb	25	75	135
1211	33	2	1	Aaaa	30	30	30

Shipping	Tax Rates	G_Total
2	1.4	136.4
2	3.0	33.0

Bentuk normal ke 2

- 2nd normal form : Tiap attribut bukan kunci harus bergantung fungsi ke attribut kunci
- Bergantung fungsi

  A→B,C,D,E

  A adalah kunci, sedangkan B,C,D,E bergantung fungsi terhadap A
- Menjadikan pengelompokan dalam tabel-tabel yang relevan

Bentuk normal ke 2

Dirubah ke bentuk

Tabel Customer

Nm_Cust*	Add_Cust	City_Cust	State_Cust	Zip	Phone_Cust	Date

Tabel Transaksi

Tabel Barang

Invoice No	Order No	Rep	FOB	QTY	Code**	Total	Sub total	Shipping	Tax Rates	G_Total	Nm_Cust**

Tabel Barang

Code*	Description	Unit Price

Bentuk normal ke 3

- Menguji bentuk normal ke 2 untuk menghilangkan ketergantungan transitif
- Ketergantungan transitif

  A→B,C,D,E

  C→E

  C adalah trans untuk E dan A, maka harus di dekomposisi menjadi :

  A→B,C,D dan C→E

Bentuk Normal ke 3

- Bisa jadi bentuk normal ke 2 juga memenuhi persyaratan normal ke 3
- Bentuk normal ke 3 bisa diaplikasikan dalam pemrograman.

Definisi Normal Ketiga

- Memenuhi bentuk 2 NF (normal kedua)

- Atribut bukan kunci tidak memiliki dependensi transitif terhadap kunci/ primary key.

Contoh relasi bentuk 2 NF, tetapi tidak memenuhi 3 NF

No Pesanan	No Urut	Kode Item	Nama Item
50001	0001	P1	Pensil
50001	0002	P2	Buku Tulis
50001	0003	P3	Penggaris
50001	0004	P4	Penghapus
50002	0001	P3	Penggaris
50002	0002	P5	Pulpen
50002	0003	P6	Spidol
50003	0001	P1	Pensil
50003	0002	P2	Buku Tulis

Penjelasan

- Atribut No Pesanan & No Urut adalah kunci primer.
- Kode item & nama item mempunyai dependensi fungsional terhadap kunci primer tersebut.
- Pada relasi di atas, setiap kode item sama, maka nilai item juga sama, terlihat adanya dependensi dua atribut itu, tetapi manakah yang menentu-kan ?

Penjelasan

- Apakah kode item bergantung pada nama item, atau sebaliknya ?
- Nama item memiliki dependensi fungsional terhadap Kode item.
- Pada relasi ini, terlihat bahwa nama item tidak memiliki dependensi secara langsung dengan kunci primer (No pesanan & No Urut). Maka Nama Item memiliki dependensi transitif terhadap kunci primer.

Bentuk 2NF menjadi 3 NF

- Relasi tersebut di dekomposisi menjadi 2 buah relasi sebagai berikut :

- Relasi Pesanan_barang
- Relasi Barang

Relasi pesanan_barang memenuhi 3 NF

No Pesanan	No Urut	Kode Item
50001	0001	P1
50001	0002	P2
50001	0003	P3
50001	0004	P4
50002	0001	P3
50002	0002	P5
50002	0003	P6
50003	0001	P1
50003	0002	P2

Relasi barang memenuhi 3 NF

Kode Item	Nama Item
P1	Pensil
P2	Buku tulis
P3	Penggaris
P4	Penghapus
P5	Pulpen
P6	Spidol

Bentuk Normal Boyce-Codd (BCNF)

Definisi :

- Memenuhi bentuk 3 NF (normal ketiga)
- Semua penentu (determinan) adalah kunci kandidat (atribut yang bersifat unik).
- BCNF adalah bentuk normal sebagai perbaikan terhadap 3 NF.
- Suatu relasi BCNF selalu memenuhi 3NF, tetapi tidak sebaliknya.

Bentuk Normal Boyce-Codd (BCNF)

- Relasi yang memenuhi 3 NF belum tentu memenuhi BCNF, karena bentuk 3 NF masih mungkin terjadi anomali.

- Contoh berikut, terdapat tabel seminar, kunci primer adalah no_siswa + seminar dengan pengertian :

Bentuk Normal Boyce-Codd (BCNF)

- Siswa dapat mengambil satu atau dua seminar
- Setiap seminar membutuhkan 2 instruktur.
- Setiap siswa dibimbing oleh salah satu dari 2 instruktur seminar.
- Setiap instruktur boleh hanya mengambil satu seminar saja.
- Pada contoh ini, no_siswa dan seminar menunjukkan seorang instruktur.

Relasi Seminar

No_Siswa	Seminar	Instruktur
2201001	2281	Budi
2201002	2281	Kardi
2201003	2291	Joni
2201002	2291	Rahmad
2201004	2291	Rahmad

Bentuk Normal Boyce-Codd (BCNF)

- Tabel seminar memenuhi bentuk 3 NF, tapi tidak BCNF karena nomor seminar masih bergantung fungsi pada instruktur.
- Jika tiap instruktur dapat mengajar hanya pada satu seminar.
- Seminar bergantung fungsi pada satu atribut bukan superkey seperti yang disyaratkan BCNF.
- Relasi seminar didekomposisi 2 relasi : relasi pengajar & relasi seminar_instruktur.

Relasi Pengajar

Instruktur	Seminar
Budi	2281
Kardi	2291
Joni	2291
Rahmad	2291

Relasi Seminar_Instruktur

No_Siswa	Instruktur
2201001	Budi
2201002	Kardi
2201003	Joni
2201002	Rahmad
2201004	Rahmad

BAB 9

STRUCTURED QUERY LANGUAGE (SQL)

- SQL merupakan singkatan dari Structure Query Language
- Dalam bahasa Inggris sering dibaca sebagai SEQUEL
- SQL merupakan bahasa query standar yang digunakan untuk mengakses basis data relational
- Penggunaan SQL pada DBMS cukup luas, SQL dapat dipakai berbagai kalangan seperti DBA, Programmer ataupun end user

- SQL sebagai bahasa administrasi basis data

  SQL dipakai oleh DBA untuk menciptakan serta mengendalikan pengaksesan basis data

- SQL sebagai bahasa query interaktif

  pengguna dapat memberikan perintah perintah untuk mengakses basis data yang sesuai dengan kebutuhannya pada saat-saat tertentu

SQL

Sebagai data sublanguage (DSL) terdiri atas:

- Data Definition Language (DDL)

  merupakan kelompok perintah yang berfungsi untuk mendefinisikan atribut atribut basis data, batasan atribut serta hubungan antar tabel. Perintah terdiri dari
  - CREATE
  - DROP
  - ALTER

SQL

Sebagai data sublanguage (DSL) terdiri atas:

- Data Manipulation Language (DML)

  kelompok perintah yang berfungsi untuk memanipulasi data dalam basis data, misalnya pengambilan, penyisipan, perubahan dah penhapusan data

  - SELECT
  - UPDATE
  - INSERT
  - DELETE

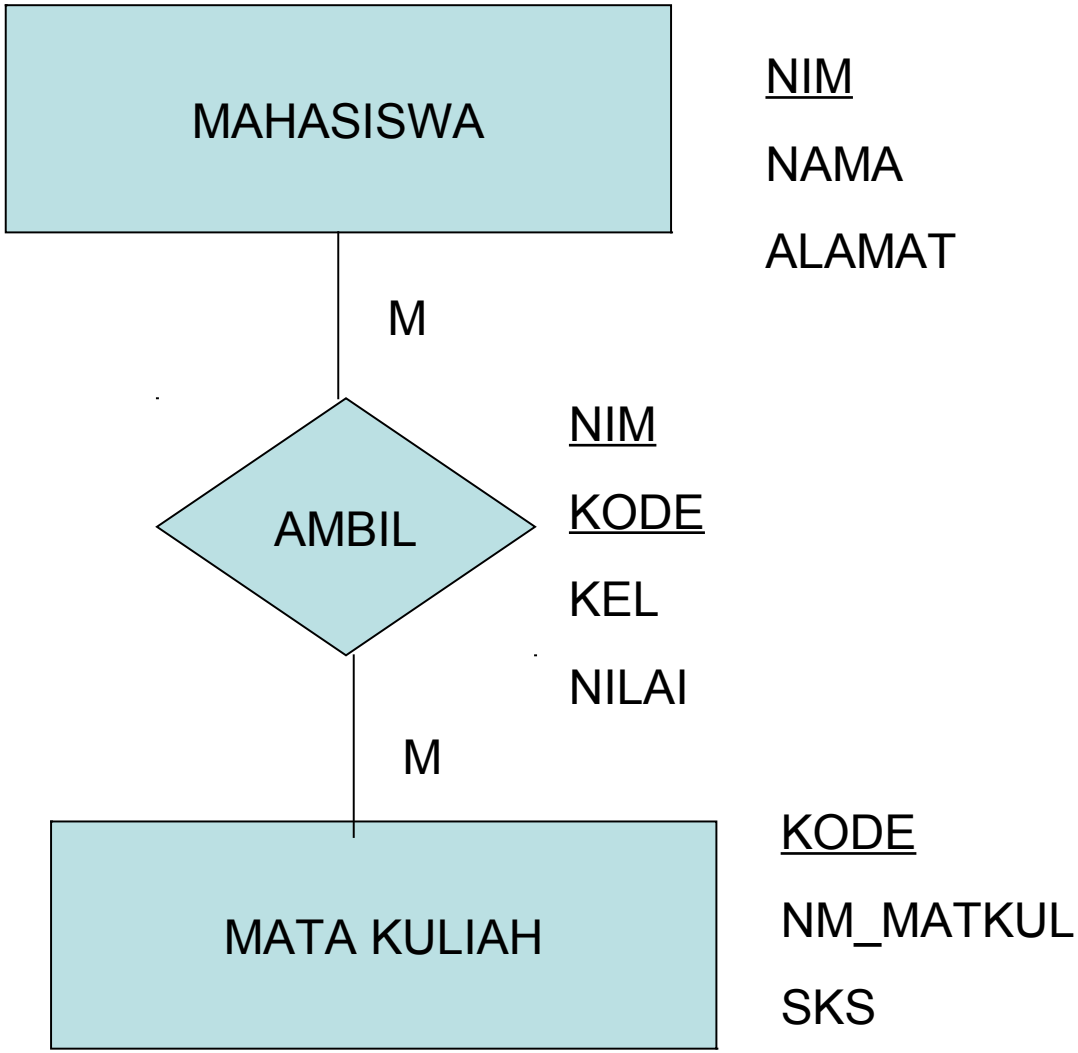
Membuat Relasi baru

Bentuk Umum CREATE TABLE

CREATE TABLE base-table
(column – definition
[,column-definition]…
[,primary-key-definition]
[,foreign-key-definition
[,foreign-key-definition]…]])

Untuk ‘column-definition’ mempunyai bentuk:

column-name data-type [NOT NULL]


MAHASISWA
NIM
NAMA
ALAMAT
M
AMBIL
NIM
KODE
KEL
NILAI
M
MATA KULIAH
KODE
NM_MATKUL
SKS

Tabel MAHASISWA

NIM	NAMA	ALAMAT
0311500012	AHMAD	JAKARTA
0322500023	RINA	CILEDUG
0322500045	ANI	JAKARTA

Tabel KULIAH

NIM	KODE	KEL	NILAI
0311500012	KP124	AA	80
0322500023	KP124	AA	50
0311500012	KP125	AB	60

Tabel MATA KULIAH

KODE	NM_MATKUL	SKS
KP124	SBD 1	2
KP125	SBD 2	3
KP126	PBD	3

Contoh CREATE TABLE

```
CREATE TABLE MAHASISWA
    (NIM CHAR(10) NOT NULL,
     NAMA CHAR(40),
     PRIMARY KEY (NIM));

CREATE TABLE MATKUL
    (KODE CHAR(5) NOT NULL,
     NM_MATKUL CHAR(40),
     SKS INTEGER,
     PRIMARY KEY (KODE));
```

```
CREATE TABLE KULIAH
    (NIM CHAR(10) NOT NULL,
     KODE CHAR(5) NOT NULL,
    KEL CHAR(2),
    NILAI INTEGER,
     PRIMARY KEY (NIM, KODE)
    FOREIGN KEY NIM
    REFERENCES MAHASISWA
    FOREIGN KEY KODE
    REFERENCES MATKUL;
```

Menambah Atribut

Bentuk Umum ALTER TABLE

```
ALTER TABLE base-table
    ADD column data type;
```

Contoh:

```
ALTER TABLE MAHASISWA ADD TGLLAHIR
    DATE;
```

Menghapus Atribut

Bentuk Umum ALTER TABLE

ALTER TABLE base-table

DROP column data type;

Contoh:

ALTER TABLE MAHASISWA DROP TGLLAHIR ;

Menghapus Relasi

Bentuk Umum DROP TABLE

DROP TABLE base – table;

Contoh:

DROP TABLE MAHASISWA;

Membuat Index

- INDEX DAPAT MEMPERCEPAT PENCARIAN DATA
- INDEKS MEMPERLAMBAT PROSES PENAMBAHAN DAN PENGHAPUSAN BARIS, KARENA SAAT TERJADI PENAMBAHAN ATAU PENGHAPUSAN BARIS, INDEKS PERLU DIPERBAHARUI

Membuat Index

Bentuk Umum CREATE INDEX

CREATE [UNIQUE] INDEX NAMA_INDEX
ON NAMA_TABLE
(column[order][,column][order]]…);

Contoh:

CREATE INDEX NIM_MHS ON MAHASISWA (NIM)

Menghapus Index

Untuk menghapus index :

DROP INDEX NAMA_INDEX ON NAMA_TABEL

CONTOH

DROP INDEX NIM_MHS ON MAHASISWA

Data Manipulation Language (DML)

- UPDATE
  - Mengubah isi satu atau beberapa atribut dari suatu tabel
- INSERT
  - Menambah satu atau beberapa baris nilai baru ke dalam suatu tabel
- DELETE
  - Menghapus sebagian atau seluruh isi dari suatu tabel
- SELECT
  - Menampilkan sebagian atau seluruh isi dari suatu tabel
  - Menampilkan kombinasi isi dari beberapa tabel

Menambah record

Bentuk Umum INSERT

```
INSERT
    INTO table [ (field [,field]…)]
    VALUES (constant [,constant]…);
```

Atau

```
INSERT
    INTO table [ (field [,field]…)]
        SELECT … FROM … WHERE …;
```

Contoh-contoh INSERT

- Single – record Insert

```
INSERT INTO MATKUL (KODE,
NM_MATKUL, SKS)
VALUES (‘KP127’, ’JARKOM’, 3);
```

Mengubah record

Bentuk Umum UPDATE

```
UPDATE table
    SET field = expression
    [, field = expression]…
    [WHERE predicate];
```

Contoh-contoh UPDATE

- Single record UPDATE

```
UPDATE MATKUL
    SET SKS = '2'
    WHERE KODE = 'KP125';
```

- Multiple record UPDATE

```
UPDATE MAHASISWA
    SET ALAMAT = 'CIPULIR'
    WHERE ALAMAT = 'JAKARTA';
```

Menghapus record

Bentuk Umum DELETE

```
DELETE FROM table
    [WHERE predicate];
```

Contoh-contoh Delete

- Single – record delete

```
DELETE FROM MATKUL
    WHERE KODE = 'KP127';
```

- Multiple record delete

```
DELETE FROM KULIAH
    WHERE NILAI < 50;
```

- Multiple record delete

```
DELETE FROM MATKUL;
```

SELECT

Bentuk umum perintah SELECT

SELECT [DISTINCT] field(s)

FROM table(s)

[WHERE predicate]

[GROUP BY field(s) [HAVING predicate]]

[ORDER BY field(s)];

Contoh Query Sederhana (1)

- Retrieval Sederhana

  Tampilkan SEMUA DARI TABEL MAHASISWA

  SELECT * FROM MAHASISWA;

  TAMPILKAN NAMA DAN ALAMAT DARI TABEL MAHASISWA URUT BERDASARKAN NAMA

  SELECT NAMA, ALAMAT FROM MAHASISWA ORDER BY NAMA

Contoh Query Sederhana (1)

- Retrieval Sederhana

  Tampilkan NIM MAHASISWA DARI TABEL KULIAH

  ```
  SELECT NIM FROM KULIAH;
  ```

  Untuk menghilangkan kemungkinan duplikasi pada hasil, maka query diatas menjadi:

  ```
  SELECT DISTINCT NIM FROM KULIAH;
  ```

ELEMEN DASAR SQL

- KONSTANTA

  Merupakan nilai yang tetap. Beberapa contoh konstanta
  - Konstanta numerik : 20, -25
  - Konstanta string : ‘Jakarta’
  - Konstanta yang disediakan SQL : sysdate (tanggal sistem) , user (nama pengguna SQL)

- Misal : tampilakan semua data mhs yang beralamat di jakarta.

```
select *   from mahasiswa
where alamat = 'jakarta'
```

NIM	NAMA	ALAMAT	UMUR
0311500012	AHMAD	JAKARTA	20
0322500023	RINA	CILEDUG	21
0322500045	TIKA	JAKARTA	22

ELEMEN DASAR SQL

- EKSPRESI

  Ekspresi adalah segala sesuati yang memberikan nilai

  simbol ekspresi yang digunakan dalam SQL

Simbol	Keterangan
*	Perkalian
/	Pembagian
+	Penambahan
-	pengurangan

- Misal : tampilkan hasil perkalian tarif dan lama dengan nama bayar

  select TARIF * LAMA as BAYAR

  from DETIL_TAGIHAN

NO TAGIHAN	KODE	TARIF	LAMA
001	005	5000	5
001	006	1000	10

ELEMEN DASAR SQL

- PREDICATE

  PREDICATE selalu menghasilkan value bertipe boolean yang isinya hanya TRUE atau FALSE

Predicate	Fungsi
comparison	Membandingkan 2 jenis yang setipe <, >, <>, =, <=, >=
Between	Membatasi data secara range Misal 5 <= x <= 10 Cara penullisannya x between 5 and 10
Exists	Memeriksa apakah data pada tabel ada atu tidak

ELEMEN DASAR SQL

Predicate	Fungsi
In	Memberikan batasan data
Like	Membandingkan bagian isi dari data suatu field
Is null	Memeriksa apakah field tersebut mempunyai isi/value atau tidak

Tabel MAHASISWA

NIM	NAMA	ALAMAT	UMUR
0311500012	AHMAD	JAKARTA	20
0322500023	RINA	CILEDUG	21
0322500045	ANI	JAKARTA	22

Tabel KULIAH

NIM	KODE	KEL	NILAI
0311500012	KP124	AA	80
0322500023	KP124	AA	50
0311500012	KP125	AB	60

Tabel MATA KULIAH

KODE	NM_MATKUL	SKS
KP124	SBD 1	2
KP125	SBD 2	3
KP126	PBD	3

- Tampilkan semua data mahasiswa yang mempunyai umur 21 tahun ke atas

  select * from mahasiswa where umur >= 21

- Tampilkan semua data mahasiswa yang mempunyai umur antara 20 thn hingga 21 thn

  Select * from mahasiswa where

  umur between 20 and 21

- Tampilkan semua data mahasiswa yang kuliah

  Select * from mahasiswa where exists (select * from kuliah where mahasiswa.nim = kuliah.nim)

- Tampilkan semua data mahasiswa yang kuliah

  Select * from mahasiswa where nim in (select nim from kuliah where mahasiswa.nim = kuliah.nim)

- Tampilkan semua data mahasiswa yang mempunyai nama ANI

  Select * from mahasiswa where nama like ‘ANI’

- Tampilkan semua data mahasiswa yang mempunyai nama HURUF DEPAN A

  Select * from mahasiswa where nama like ‘A%’

- Tampilkan data mahasiswa yang data umurnya kosong

  select * from mahasiswa where umur is null

ELEMEN DASAR SQL

- OPERATOR LOGIKA

Predicate	Fungsi
AND	Membandingkan 2 predicate, hanya akan benar jika 2 predicate bernilai benar, selainnya salah
OR	Membandingkan 2 predicate, hanya akan salah jika 2 predicate bernilai salahm selaiinya benar
NOT	Membalikkan nilai predicate

Tabel MAHASISWA

NIM	NAMA	ALAMAT	UMUR
0311500012	AHMAD	JAKARTA	20
0322500023	RINA	CILEDUG	21
0322500045	ANI	JAKARTA	22

Tabel KULIAH

NIM	KODE	KEL	NILAI
0311500012	KP124	AA	80
0322500023	KP124	AA	50
0311500012	KP125	AB	60

Tabel MATA KULIAH

KODE	NM_MATKUL	SKS
KP124	SBD 1	2
KP125	SBD 2	3
KP126	PBD	3

- Tampilkan semua data mahasiswa yang mempunyai umur 21 tahun ke atas dan alamat nya di jakarta

  select * from mahasiswa where umur >= 21 and alamat='jakarta'

- Tampilkan semua data mahasiswa yang mempunyai umur 21 tahun ke atas atau 20 tahun ke bawah

  Select * from mahasiswa where

  umur > 21 or umur < 20

- Tampilkan semua data mahasiswa yang tidak kuliah

  Select * from mahasiswa where nim not in (select distinct(nim) from kuliah where mahasiswa.nim=kuliah.nim)

Fungsi – fungsi Aggregate

- COUNT

  Banyaknya nilai-nilai pada satu kolom
- SUM

  Jumlah nilai dari satu kolom
- AVG

  Rata-rata nilai dari satu kolom
- MAX

  Nilai terbesar yang ada pada satu kolom
- MIN

  Nilai terkecil yang ada pada satu kolom