TUGAS

1. Jelaskan apa yang dimaksud dengan normalisasi database?

JAWABAN: Normalisasi data berfungsi untuk menciptakan data yang tidak redundan dan terstruktur sehingga dapat dengan mudah dikelompokkan, dianalisis, dan dimodifikasi supaya menghasilkan data yang seragam. Proses normalisasi database mencakup membersihkan atau mengatur data agar lebih terstruktur sesuai standar. Dalam normalisasi, tabel-tabel dalam database diatur agar setiap tabel hanya mengandung informasi yang terkait langsung dengan satu entitas tertentu.

2. Apa yang menjadi tujuan pembuatan database perlu dinormalisasikan?

- **Mengurangi Redundansi Data:** Menghindari penyimpanan data berulang, menghemat ruang, dan meningkatkan efisiensi.
- **Meningkatkan Integritas dan Konsistensi:** Data disimpan di satu tempat, memudahkan pembaruan tanpa inkonsistensi.
- **Menghindari Anomali Database:** Mencegah masalah penyisipan, penghapusan, dan pembaruan yang bisa menyebabkan kehilangan atau ketidaksesuaian data.
- **Memudahkan Pemeliharaan:** Database lebih mudah dikelola dan diperbarui tanpa mengganggu struktur data.
- **Meningkatkan Kinerja Query:** Struktur yang efisien mempercepat eksekusi query karena data tidak berlebih.

3. Uraikan proses tingkat tentang normalisasi dari tahap 1NF hingga 5NF!

1️⃣ First Normal Form (1NF) – Hilangkan Repeating Groups

Contoh Data Awal (Belum 1NF)

ID Pelanggan	Nama Pelanggan	Alamat	Produk Dibeli	Harga
201	Ali	Jakarta	Laptop, Mouse	10.000.000, 150.000

ID Pelanggan	Nama Pelanggan	Alamat	Produk Dibeli	Harga
202	Rina	Bandung	Printer	2.500.000
203	Budi	Surabaya	Laptop, Printer	10.000.000, 2.500.000

Masalah:

- Kolom "Produk Dibeli" berisi lebih dari satu nilai, yang tidak sesuai dengan 1NF.
- Sulit dikelola jika pelanggan membeli lebih dari satu produk.
- Jika harga produk berubah, harus mengedit banyak data secara manual.

Solusi: Memenuhi 1NF dengan Pemisahan Tabel

Tabel Pelanggan

ID Pelanggan	Nama Pelanggan	Alamat
201	Ali	Jakarta
202	Rina	Bandung
203	Budi	Surabaya

Tabel Produk

ID Produk	Nama Produk	Harga
P001	Laptop	10.000.000
P002	Mouse	150.000
P003	Printer	2.500.000

Tabel Relasi Pelanggan - Produk (Transaksi)

ID Pelanggan	ID Produk
201	P001
201	P002
202	P003
203	P001
203	P003

✅ **Setiap kolom hanya memiliki satu nilai (Atomicity).** ✅ **Data lebih mudah diubah dan dikelola.**

2️⃣ Second Normal Form (2NF) – Hilangkan Ketergantungan Parsial

Syarat:

✅ Sudah memenuhi **1NF**. ✅ Semua kolom **harus bergantung sepenuhnya pada primary key (PK)**. ✅ Jika ada kolom yang hanya bergantung pada sebagian dari PK, harus dipisahkan ke tabel baru.

Masalah di 1NF:

- Primary Key pada **Tabel Transaksi** adalah **(ID Pelanggan, ID Produk)**, tetapi ada kemungkinan **data pelanggan seperti "Alamat" masih tergantung pada ID Pelanggan**, bukan ID Produk.

Solusi:

- Pastikan **setiap kolom non-kunci hanya bergantung pada kunci utama secara keseluruhan, bukan sebagian.**
- Karena **"Alamat" hanya bergantung pada ID Pelanggan**, kita pisahkan agar tidak ada **partial dependency**.

✅ **Hasil 2NF (Tidak ada ketergantungan sebagian)**

- **Tabel Pelanggan (Tidak berubah)**
- **Tabel Produk (Tidak berubah)**
- **Tabel Transaksi (Tidak berubah, hanya memastikan tidak ada atribut tambahan)**

3️⃣ Third Normal Form (3NF) - Menghilangkan Transitive Dependency

Masalah di 2NF: Misalnya, dalam **Tabel Produk**, kita menambahkan kolom "Kategori Produk":

ID Produk	Nama Produk	Harga	Kategori
P001	Laptop	10.000.000	Elektronik
P002	Mouse	150.000	Aksesoris
P003	Printer	2.500.000	Elektronik

Solusi:

- Buat tabel baru untuk **Kategori Produk**.

✅ **Hasil 3NF (Tidak ada transitive dependency)**

Tabel Kategori Produk

ID Kategori	Nama Kategori
K01	Elektronik
K02	Aksesoris

Tabel Relasi Produk-Kategori

ID Produk	ID Kategori
P001	K01
P002	K02
P003	K01

4️⃣ Boyce-Codd Normal Form (4CNF) - Memastikan Ketergantungan Fungsional

Masalah di 3NF: Jika ada **lebih dari satu kandidat kunci**, kita harus memastikan bahwa **semua determinan adalah superkey**.

Misalnya, dalam **Tabel Transaksi** kita menambahkan **"Metode Pembayaran"**, tetapi **satu pelanggan bisa menggunakan beberapa metode pembayaran untuk produk yang sama**.

ID Pelanggan	ID Produk	Metode Pembayaran
201	P001	Kartu Kredit
201	P001	Transfer Bank
202	P003	Tunai

✅ **Hasil BCNF**

Tabel Metode Pembayaran

ID Pelanggan	ID Produk	Metode Pembayaran
201	P001	Kartu Kredit
201	P001	Transfer Bank
202	P003	Tunai

5️⃣ Fourth Normal Form (5NF) - Menghilangkan Multivalued Dependency

Misalnya, satu **pelanggan bisa memiliki banyak alamat pengiriman** untuk transaksi yang sama:

ID Pelanggan	ID Produk	Alamat Pengiriman
201	P001	Jakarta
201	P001	Bandung

✅ **Hasil 4NF**

Tabel Alamat Pengiriman

ID Pelanggan	Alamat Pengiriman
201	Jakarta
201	Bandung

Dengan menerapkan 1NF hingga 4NF, database menjadi lebih terstruktur dan efisien.

4. Buatlah contoh normalisasikan database minimal dari tahap 1NF hingga 3NF pada kasus database proyek basis data sekolah masing-masing!

Tahap 1NF (First Normal Form)

Syarat 1NF:

1. Setiap kolom harus memiliki nilai atomic (tidak ada nilai yang berulang dalam satu sel).
2. Setiap tabel memiliki primary key yang unik.
3. Tidak boleh ada grup nilai berulang dalam satu kolom.
   Contoh tabel sebelum 1NF:

id_transaksi	id_siswa	nama_siswa	tanggal_transaksi	produk (gabungan)	jumlah (gabungan)	t
1	101	Budi	2024-02-27	Nasi Goreng, Es Teh	1, 2	
2	102	Siti	2024-02-27	Ayam Bakar	1	

Masalah:

- Kolom `produk` dan `jumlah` menyimpan lebih dari satu nilai, yang melanggar **1NF**.

Solusi 1NF

Pisahkan produk dan jumlah ke dalam tabel **detail_transaksi**.

Tabel transaksi (Setelah 1NF)

id_transaksi	id_siswa	tanggal_transaksi
1	101	2024-02-27
2	102	2024-02-27

Tabel detail_transaksi (Setelah 1NF)

id_detail	id_transaksi	id_produk	jumlah	total_harga
1	1	201	1	10.00
2	1	202	2	5.00
3	2	203	1	10.00

Tabel produk

id_produk	nama_produk	harga	id_kategori
201	Nasi Goreng	10.00	1
202	Es Teh	2.50	2
203	Ayam Bakar	10.00	1

Tahap 2NF (Second Normal Form)

Syarat 2NF:

1. Sudah dalam 1NF.
2. Setiap kolom **non-key** harus bergantung **sepenuhnya** pada **primary key**, bukan hanya sebagian dari primary key.

Masalah:

- Kolom `total_harga` di tabel `detail_transaksi` bisa dihitung dari **jumlah × harga produk**, menyebabkan redundansi data.
- `nama_kategori` sebaiknya tidak ada di tabel `produk`, karena bergantung pada `id_kategori`.

Solusi 2NF

1. **Hilangkan** `total_harga` **dari** `detail_transaksi`, karena bisa dihitung langsung.
2. **Pisahkan kategori produk** ke dalam tabel `kategori`.

Tabel detail_transaksi (Setelah 2NF)

id_detail	id_transaksi	id_produk	jumlah
1	1	201	1
2	1	202	2
3	2	203	1

Tabel kategori (Baru setelah 2NF)

id_kategori	nama_kategori
1	Makanan
2	Minuman

Tahap 3NF (Third Normal Form)

Syarat 3NF:

1. Sudah dalam 2NF.
2. Tidak ada **ketergantungan transitif** (Kolom non-key tidak boleh tergantung pada kolom non-key lainnya).

Masalah:

- **Saldo siswa tidak perlu disimpan langsung di tabel** `siswa`, karena bisa dihitung dari histori transaksi.
- **total_bayar dalam** `transaksi` **bisa dihitung dari jumlah pembelian dalam** `detail_transaksi`, jadi tidak perlu disimpan langsung.

Solusi 3NF

1. **Hilangkan** `saldo` **dari tabel** `siswa`, karena bisa dihitung dari histori transaksi.
2. **Hilangkan** `total_bayar` **dari** `transaksi`, karena bisa dihitung dari jumlah pembelian.

Tabel siswa (Setelah 3NF)

id_siswa	nama_siswa	kelas
101	Budi	X IPA

id_siswa	nama_siswa	kelas
102	Siti	XI IPS

Tabel transaksi (Setelah 3NF)

id_transaksi	id_siswa	tanggal_transaksi
1	101	2024-02-27
2	102	2024-02-27

Hasil Akhir Database dalam 3NF

1. Tabel `siswa`

- **PK**: `id_siswa`
- **Kolom**: `nama_siswa`, `kelas`

2. Tabel `transaksi`

- **PK**: `id_transaksi`
- **FK**: `id_siswa`
- **Kolom**: `tanggal_transaksi`

3. Tabel `detail_transaksi`

- **PK**: `id_detail`
- **FK**: `id_transaksi`, `id_produk`
- **Kolom**: `jumlah`

4. Tabel `produk`

- **PK**: `id_produk`
- **FK**: `id_kategori`
- **Kolom**: `nama_produk`, `harga`, `stock`

5. Tabel `kategori`

- **PK**: `id_kategori`
- **Kolom**: `nama_kategori`

Kesimpulan

- **1NF:** Menghapus grup data berulang dengan memisahkan `detail_transaksi`.
- **2NF:** Memisahkan kategori dari `produk` dan menghilangkan kolom yang bisa dihitung.

- **3NF:** Menghilangkan **kolom yang bisa dihitung** untuk mengurangi redundansi.