Perancangan Sistem Basis Data Relasional untuk Studio Kebugaran

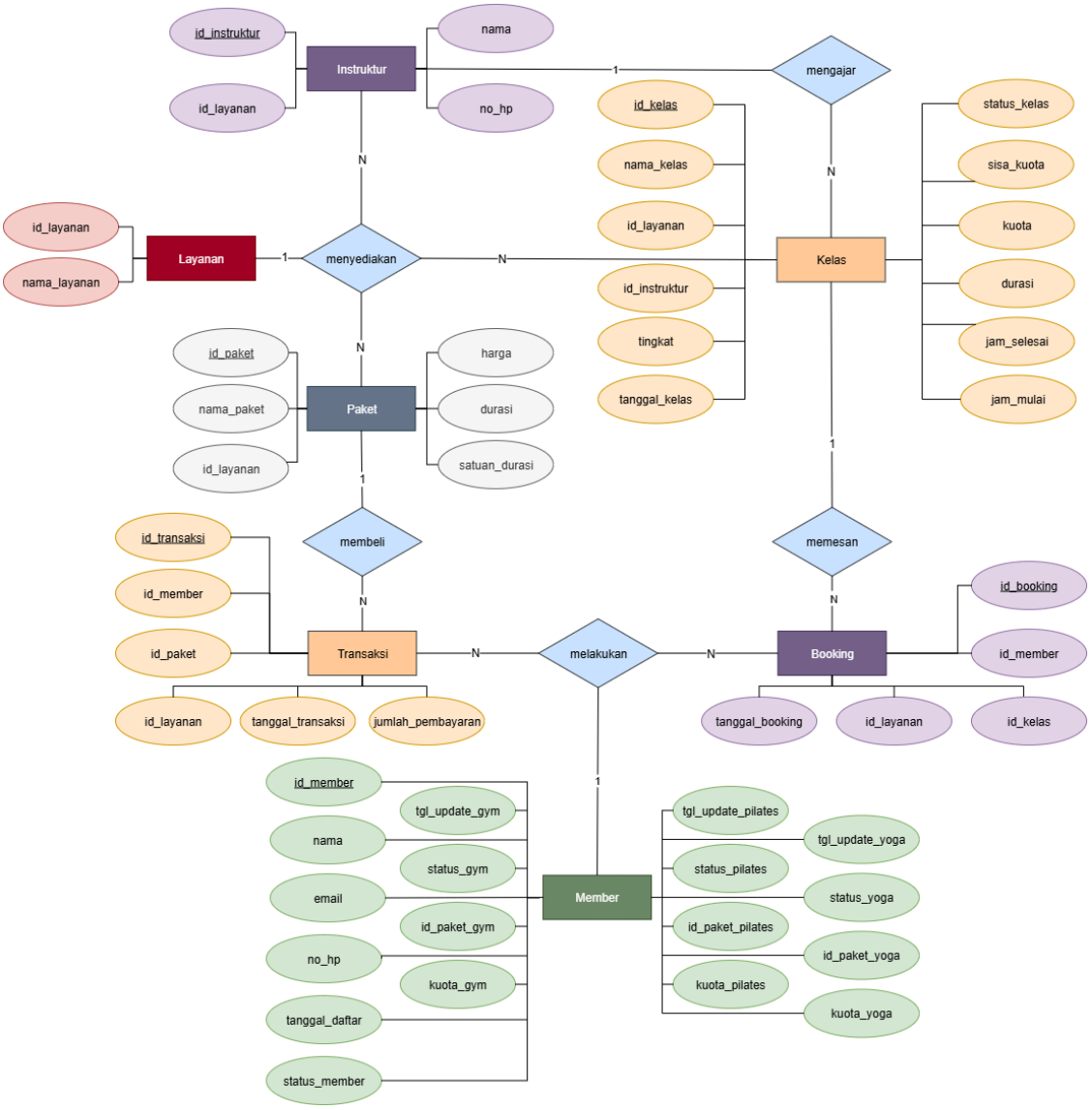
Studio kebugaran seperti yoga, pilates, dan gym kini semakin diminati seiring meningkatnya kesadaran masyarakat akan gaya hidup sehat. Pengelolaan aktivitas seperti jadwal kelas, booking member, dan layanan instruktur memerlukan sistem yang efisien agar operasional berjalan lancar.

Penggunaan basis data relasional memungkinkan data dikelola secara terstruktur dan terintegrasi. Dengan bantuan SQL, pengolahan data menjadi lebih cepat, akurat, dan mudah dianalisis. Oleh karena itu, dibutuhkan perancangan sistem basis data berbasis SQL untuk mendukung manajemen layanan di studio kebugaran.

Tujuan implementasi SQL untuk perancangan sistem basis data adalah untuk mengelola data penting seperti member, instruktur, kelas, jadwal, booking, dan paket layanan, serta mempermudah akses, pencarian, dan analisis data operasional studio.

Struktur Basis Data

Entity-Relationship Diagram



Sebuah studio kebugaran memiliki beberapa layanan yaitu gym, pilates, dan yoga. Setiap layanan menyediakan beberapa paket langganan, instruktur, dan kelas. Setiap instruktur dapat mengajar beberapa kelas, namun setiap kelas tidak dapat diajar oleh lebih dari satu instruktur.

Setiap member dapat melakukan transaksi pembelian paket dan pemesanan kelas, di mana setiap transaksi hanya dapat digunakan untuk satu pembelian paket dan setiap pemesanan hanya dapat digunakan untuk memesan satu kelas. Setiap paket dapat dibeli melalui beberapa transaksi (atau oleh beberapa member), begitupun dengan setiap kelas dapat dipesan melalui beberapa pemesanan (atau oleh beberapa member).

Tabel dan Atribut

Nama Kolom	Tipe Data	Deskripsi	Keterangan Tambahan
TABEL LAYANAN			
id_layanan	INT	Nilai unik berupa identitas dari layanan yang disediakan.	PRIMARY KEY, AUTO INCREMENT, NOT NULL
nama_layanan	VARCHAR	Nama layanan yang disediakan (Gym/Pilates/Yoga)	NOT NULL
TABEL PAKET			
id_paket	INT	Nomor identifikasi unik berupa identitas dari paket yang disediakan.	PRIMARY KEY, AUTO INCREMENT, NOT NULL
nama_paket	VARCHAR	Nama paket yang disediakan.	NOT NULL
id_layanan	INT	Nomor identifikasi untuk jenis layanan paket.	FOREIGN KEY, NOT NULL
harga	DECIMAL	Harga yang perlu dibayarkan pelanggan ketika melakukan transaksi pembelian paket.	NOT NULL
durasi	INT	Durasi paket aktif sejak pembelian paket.	NOT NULL
satuan_durasi	ENUM	Satuan durasi paket aktif (hari/sesi).	NOT NULL
TABEL INSTRUKTUR			
id_instruktur	INT	Nomor identifikasi unik berupa identitas dari instruktur.	PRIMARY KEY, AUTO INCREMENT, NOT NULL

id_layanan	INT	Nomor idenifikasi untuk jenis layanan yang disediakan oleh instruktur.	FOREIGN KEY, NOT NULL
nama	VARCHAR	Nama instruktur.	NOT NULL
no_hp	VARCHAR	Nomor telepon instruktur.	NOT NULL
TABEL KELAS			
id_kelas	INT	Nomor identifikasi unik berupa identitas kelas.	PRIMARY KEY, AUTO INCREMENT, NOT NULL
nama_kelas	VARCHAR	Nama kelas.	NOT NULL
id_layanan	INT	Nomor identifikasi untuk layanan yang menyediakan kelas.	FOREIGN KEY, NOT NULL
id_instruktur	INT	Nomor identifikasi untuk instruktur yang mengajar kelas.	FOREIGN KEY, NOT NULL
tingkat	ENUM	Tingkat kesulitan kelas (beginner/intermediate/advance).	NOT NULL
tanggal_kelas	DATE	Tanggal kelas dilaksanakan.	NOT NULL
jam_mulai	TIME	Jam kelas dimulai.	NOT NULL
jam-selesai	TIME	Jam kelas selesai.	NOT NULL
durasi	INT	Durasi atau lama watu kelas berlangsung.	
kuota	INT	Kuota member yang dapat mengikuti kelas.	NOT NULL
sisa_kuota	INT	Sisa kuota kelas yang tersedia.	NOT NULL
status_kelas	ENUM	Status kelas (tersedia/kuota penuh/selesai).	DEFAULT ‘TERSEDIA’
TABEL MEMBER			
id_member	INT	Nomor identifikasi unik berupa identitas dari member.	PRIMARY KEY, AUTO INCREMENT, NOT NULL

nama	VARCHAR	Nama member.	NOT NULL
email	VARCHAR	Alamat e-mail member.	UNIQUE, NOT NULL
no_hp	VARCHAR	Nomor telepon member.	UNIQUE, NOT NULL
tanggal_daftar	DATE	Tanggal pendaftaran member.	NOT NULL
status_member	ENUM	Status member (aktif/nonaktif) yang menunjukkan apakah member memiliki paket yang aktif.	DEFAULT 'NONAKTIF'
tgl_update_gym	DATE	Tanggal terakhir kali member melakukan pembelian paket layanan gym.	
status_gym	ENUM	Status paket untuk layanan gym (aktif/nonaktif).	DEFAULT 'NONAKTIF'
id_paket_gym	INT	Nomor identifikasi paket layanan gym yang dibeli oleh member.	FOREIGN KEY
kuota_gym	INT	Sisa kuota layanan gym yang dimiliki oleh member sebelum status paket nonaktif.	
tgl_update_pilates	DATE	Tanggal terakhir kali member melakukan pembelian paket layanan pilates.	
status_pilates	ENUM	Status paket untuk layanan pilates (aktif/nonaktif).	DEFAULT 'NONAKTIF'
id_paket_pilates	INT	Nomor identifikasi paket layanan pilates yang dibeli oleh member.	FOREIGN KEY
kuota_pilates	INT	Sisa kuota layanan pilates yang dimiliki oleh member sebelum status paket nonaktif.	
tgl_update_yoga	DATE	Tanggal terakhir kali member melakukan pembelian paket layanan yoga.	
status_yoga	ENUM	Status paket untuk layanan yoga (aktif/nonaktif).	DEFAULT 'NONAKTIF'

id_paket_yoga	INT	Nomor identifikasi paket layanan yoga yang dibeli oleh member.	FOREIGN KEY
kuota_yoga	INT	Sisa kuota layanan yoga yang dimiliki oleh member sebelum status paket nonaktif.	
TABEL TRANSAKSI			
id_transaksi	INT	Nomor identifikasi unik berupa identitas transaksi pembelian paket.	PRIMARY KEY, AUTO INCREMENT, NOT NULL
id_member	INT	Nomor identifikasi untuk member yang melakukan transaksi pembelian paket.	FOREIGN KEY, NOT NULL
id_paket	INT	Nomor identifikasi untuk paket yang dibeli oleh member.	FOREIGN KEY, NOT NULL
id_layanan	INT	Nomor identifikasi untuk layanan dari paket yang dibeli oleh member.	FOREIGN KEY, NOT NULL
tanggal_transaksi	DATE	Tanggal transaksi dilakukan.	NOT NULL
jumlah_pembayaran	DECIMAL	Jumlah pembayaran yang dilakukan oleh member.	NOT NULL
TABEL BOOKING			
id_booking	INT	Nomor identifikasi unik berupa identitas pemesanan kelas.	PRIMARY KEY, AUTO INCREMENT, NOT NULL
id_member	INT	Nomor identifikasi untuk member yang melakukan booking kelas.	FOREIGN KEY, NOT NULL
id_kelas	INT	Nomor identifikasi untuk kelas yang dipesan oleh member.	FOREIGN KEY, NOT NULL
id_layanan	INT	Nomor identifikasi untuk layanan dari kelas yang dipesan oleh member.	FOREIGN KEY, NOT NULL
tanggal_booking	DATE	Tanggal pemesanan dilakukan.	NOT NULL

Query

1. Membuat basis data

Pada project ini, kita akan membuat suatu basis data yang dinamakan dengan ‘studio’ dengan perintah CREATE.

```
CREATE DATABASE studio;
```

2. Menggunakan basis data
   Sebelum menjalankan perintah untuk membuat struktur basis data, kita akan menjalankan perintah USE untuk mengaktifkan basis data yang digunakan dalam sesi kerja.

```
USE studio;
```

---

3. Membuat tabel layanan

```
CREATE TABLE layanan (
    id_layanan INT PRIMARY KEY AUTO_INCREMENT,
    nama_layanan VARCHAR(50) NOT NULL
);
```

Tabel layanan pada basis data ini menyimpan informasi mengenai jenis layanan yang disediakan dalam studio yaitu gym, pilates, dan yoga. Pada tabel ini, primary key-nya adalah ‘id_layanan’ yang diatur sebagai auto increment (nilainya akan diinput dan bertambah secara otomatis seiring input data/baris baru).

4. Menambahkan data pada tabel layanan
   Maka dari itu, nilai ‘gym’, ‘pilates’, dan ‘yoga’ akan ditambahkan ke dalam tabel layanan.

```
INSERT INTO layanan(nama_layanan) VALUES
('Gym'),
('Pilates'),
('Yoga');
```

5. Menampilkan isi tabel layanan

```
SELECT * FROM layanan;
```

Maka, tabel yang akan ditampilkan adalah,

	id_layanan	nama_layanan
▶	1	Gym
	2	Pilates
	3	Yoga
*	NULL	NULL

---

6. Membuat tabel paket

```
CREATE TABLE paket (
    id_paket INT PRIMARY KEY AUTO_INCREMENT,
    nama_paket VARCHAR(100) NOT NULL,
    id_layanan INT NOT NULL,
    harga DECIMAL(12, 2) NOT NULL,
    durasi INT NOT NULL,
    satuan_durasi ENUM('hari', 'sesi') NOT NULL,
    FOREIGN KEY (id_layanan) REFERENCES layanan(id_layanan)
);
```

Tabel paket dalam basis data ini menyimpan informasi mengenai paket yang tersedia untuk setiap layanan. Pada tabel ini, primary key-nya adalah ‘id_paket’ yang diatur sebagai auto increment, sedangkan ‘id_layanan’ adalah foreign key yang merujuk pada jenis layanan yang menyediakan paket tersebut.

7. Menambahkan data pada tabel paket
   Berikut adalah contoh paket yang disediakan untuk setiap layanan,

```
INSERT INTO paket (nama_paket, id_layanan,
harga, durasi, satuan_durasi) VALUES
('Basic',     1,  750000.00, 30,  'hari'),
('Pro',       1, 2000000.00, 90,  'hari'),
('Ultimate',  1, 7500000.00, 360, 'hari'),
('Drop-in',   1,  100000.00, 1,   'sesi'),
('Basic',     2,  850000.00, 4,   'sesi'),
('Pro',       2, 1600000.00, 8,   'sesi'),
('Ultimate',  2, 2300000.00, 12,  'sesi'),
('Drop-in',   2,  200000.00, 1,   'sesi'),
('Basic',     3,  700000.00, 4,   'sesi'),
('Pro',       3, 1400000.00, 8,   'sesi'),
('Ultimate',  3, 2100000.00, 12,  'sesi'),
('Drop-in',   3,  180000.00, 1,   'sesi');
```

8. Menampilkan isi tabel paket

```
SELECT * FROM paket;
```

   Maka, tabel yang akan ditampilkan adalah,

id_paket	nama_paket	id_layanan	harga	durasi	satuan_durasi
1	Basic	1	750000.00	30	hari
2	Pro	1	2000000.00	90	hari
3	Ultimate	1	7500000.00	360	hari
4	Drop-in	1	100000.00	1	sesi
5	Basic	2	850000.00	4	sesi
6	Pro	2	1600000.00	8	sesi
7	Ultimate	2	2300000.00	12	sesi
8	Drop-in	2	200000.00	1	sesi
9	Basic	3	700000.00	4	sesi
10	Pro	3	1400000.00	8	sesi
11	Ultimate	3	2100000.00	12	sesi
12	Drop-in	3	180000.00	1	sesi

---

9. Membuat tabel instruktur

```
CREATE TABLE instruktur (
    id_instruktur INT PRIMARY KEY AUTO_INCREMENT,
    id_layanan INT NOT NULL,
    nama VARCHAR(100) NOT NULL,
    no_hp VARCHAR(20),
    FOREIGN KEY (id_layanan) REFERENCES layanan(id_layanan)
);
```

Tabel instruktur dalam basis data ini menyimpan informasi mengenai instruktur yang mengajar untuk layanan yang tersedia. Pada tabel ini, primary key-nya adalah ‘id_instruktur’ yang diatur sebagai auto increment, sedangkan ‘id_layanan’ adalah foreign key yang merujuk layanan yang diajar oleh instruktur tersebut.

10. Menambahkan data pada tabel instruktur
    Berikut adalah contoh data instruktur yang mengajar di studio,

```
INSERT INTO instruktur (nama, id_layanan, no_hp)
VALUES
('Arif Ramadhan',      1, '081212341234'),
('Dimas Hartono',      1, '081222334455'),
('Rendra Saputra',     1, '081223344556'),
('Rafi Aditya',        1, '081234567000'),
('Kevin Alfarizi',     1, '081267891234'),
('Rico Mahendra',      2, '081277788899'),
('Ayu Wulandari',      2, '081289998877'),
('Tommy Gunawan',      2, '081212123123'),
('Carissa Putri',      2, '081234123456'),
('Anita Surya',        2, '081277755555'),
('Maya Pratiwi',       3, '081298765432'),
('Sabrina Lestari',    3, '081234556677'),
('Intan Permata',      3, '081278901234'),
('Nadia Fauziah',      3, '081298765111'),
('Yoga Santoso',       3, '081289898989');
```

11. Menampilkan isi tabel instruktur

```
SELECT * FROM instruktur;
```

    Maka, tabel yang akan ditampilkan adalah,

id_instruktur	id_layanan	nama	no_hp
1	1	Arif Ramadhan	081212341234
2	1	Dimas Hartono	081222334455
3	1	Rendra Saputra	081223344556
4	1	Rafi Aditya	081234567000
5	1	Kevin Alfarizi	081267891234
6	2	Rico Mahendra	081277788899
7	2	Ayu Wulandari	081289998877
8	2	Tommy Gunawan	081212123123
9	2	Carissa Putri	081234123456
10	2	Anita Surya	081277755555
11	3	Maya Pratiwi	081298765432
12	3	Sabrina Lestari	081234556677
13	3	Intan Permata	081278901234
14	3	Nadia Fauziah	081298765111
15	3	Yoga Santoso	081289898989

---

12. Membuat tabel member

```sql
CREATE TABLE member (
    id_member INT PRIMARY KEY AUTO_INCREMENT,
    nama VARCHAR(100) NOT NULL,
    email VARCHAR(100) UNIQUE,
    no_hp VARCHAR(20) UNIQUE,
    tanggal_daftar DATE NOT NULL
);
```

Tabel member dalam basis data ini menyimpan informasi mengenai member yang terdaftar di studio. Setiap pendaftaran membutuhkan informasi pribadi member berupa nama, email, dan nomor telepon. Email dan nomor telepon diatur sebagai UNIQUE, sehingga pendaftaran member yang berbeda dengan email dan/atau nomor telepon yang sama akan gagal. Pada tabel ini, primary key-nya adalah 'id_member' yang diatur sebagai auto increment.

13. Menambahkan kolom pada tabel member

```sql
ALTER TABLE member
ADD status_member ENUM('AKTIF', 'NONAKTIF') DEFAULT 'NONAKTIF',
ADD tgl_update_gym DATE,
ADD status_gym ENUM('AKTIF', 'NONAKTIF') DEFAULT 'NONAKTIF',
ADD id_paket_gym INT,
ADD kuota_gym INT,
ADD tgl_update_pilates DATE,
ADD status_pilates ENUM('AKTIF', 'NONAKTIF') DEFAULT 'NONAKTIF',
ADD id_paket_pilates INT,
ADD kuota_pilates INT,
ADD tgl_update_yoga DATE,
ADD status_yoga ENUM('AKTIF', 'NONAKTIF') DEFAULT 'NONAKTIF',
ADD id_paket_yoga INT,
ADD kuota_yoga INT,
ADD CONSTRAINT fk_paket_gym FOREIGN KEY (id_paket_gym) REFERENCES paket(id_paket),
ADD CONSTRAINT fk_paket_pilates FOREIGN KEY (id_paket_pilates) REFERENCES paket(id_paket),
ADD CONSTRAINT fk_paket_yoga FOREIGN KEY (id_paket_yoga) REFERENCES paket(id_paket);
```

Untuk setiap layanan gym, pilates, dan yoga, akan terdapat informasi mengenai 'tgl_update_ . . .' yaitu tanggal pembaruan paket untuk layanan tersebut, 'status_ . . .' yaitu status paket untuk layanan tersebut ('AKTIF' jika terdapat paket yang sedang aktif atau default 'NONAKTIF'), 'id_paket_ . . .' yaitu jenis paket yang sedang aktif untuk layanan tersebut, dan 'kuota_ . . .' yaitu sisa kuota paket yang dapat digunakan oleh member. Kemudian, kolom 'status_member' menunjukkan apakah member memiliki setidaknya satu paket layanan yang aktif.

14. Menambahkan data pada tabel member

Berikut adalah contoh data member yang terdaftar di studio,

```sql
INSERT INTO member (nama, email, no_hp, tanggal_daftar) VALUES
('Andi Saputra', 'andi.s@example.com', '081234567801', '2025-07-01'),
('Bella Rahma', 'bella.r@example.com', '081234567802', '2025-07-01'),
('Citra Dewi', 'citra.d@example.com', '081234567803', '2025-07-01'),
('Dedi Pratama', 'dedi.p@example.com', '081234567804', '2025-07-02'),
('Eka Wulandari', 'eka.w@example.com', '081234567805', '2025-07-02'),
('Farhan Yusuf', 'farhan.y@example.com', '081234567806', '2025-07-02'),
('Gita Lestari', 'gita.l@example.com', '081234567807', '2025-07-03'),
('Hendra Kurnia', 'hendra.k@example.com', '081234567808', '2025-07-03'),
('Intan Permata', 'intan.p@example.com', '081234567809', '2025-07-03'),
('Joko Sembada', 'joko.s@example.com', '081234567810', '2025-07-04'),
('Kiki Amelia', 'kiki.a@example.com', '081234567811', '2025-07-04'),
('Lutfi Hidayat', 'lutfi.h@example.com', '081234567812', '2025-07-04'),
('Maya Kartika', 'maya.k@example.com', '081234567813', '2025-07-05'),
('Nando Firmansyah', 'nando.f@example.com', '081234567814', '2025-07-05'),
('Ovi Rizki', 'ovi.r@example.com', '081234567815', '2025-07-05');
```

15. Menampilkan isi tabel member

```sql
SELECT * FROM member;
```

Maka, tabel yang akan ditampilkan adalah,

id_member	nama	email	no_hp	tanggal_daftar
1	Andi Saputra	andi.s@example.com	081234567801	2025-07-01
2	Bella Rahma	bella.r@example.com	081234567802	2025-07-01
3	Citra Dewi	citra.d@example.com	081234567803	2025-07-01
4	Dedi Pratama	dedi.p@example.com	081234567804	2025-07-02
5	Eka Wulandari	eka.w@example.com	081234567805	2025-07-02
6	Farhan Yusuf	farhan.y@example.com	081234567806	2025-07-02
7	Gita Lestari	gita.l@example.com	081234567807	2025-07-03
8	Hendra Kurnia	hendra.k@example.com	081234567808	2025-07-03
9	Intan Permata	intan.p@example.com	081234567809	2025-07-03
10	Joko Sembada	joko.s@example.com	081234567810	2025-07-04
11	Kiki Amelia	kiki.a@example.com	081234567811	2025-07-04
12	Lutfi Hidayat	lutfi.h@example.com	081234567812	2025-07-04
13	Maya Kartika	maya.k@example.com	081234567813	2025-07-05
14	Nando Firman...	nando.f@example.com	081234567814	2025-07-05
15	Ovi Rizki	ovi.r@example.com	081234567815	2025-07-05

status_member	tgl_update_gym	status_gym	id_paket_gym	kuota_gym	tgl_update_pilates	status_pilates
NONAKTIF	NULL	NONAKTIF	NULL	NULL	NULL	NONAKTIF
NONAKTIF	NULL	NONAKTIF	NULL	NULL	NULL	NONAKTIF
NONAKTIF	NULL	NONAKTIF	NULL	NULL	NULL	NONAKTIF
NONAKTIF	NULL	NONAKTIF	NULL	NULL	NULL	NONAKTIF
NONAKTIF	NULL	NONAKTIF	NULL	NULL	NULL	NONAKTIF
NONAKTIF	NULL	NONAKTIF	NULL	NULL		NONAKTIF
NONAKTIF	NULL	NONAKTIF	NULL	NULL	NULL	NONAKTIF
NONAKTIF	NULL	NONAKTIF	NULL	NULL	NULL	NONAKTIF
NONAKTIF	NULL	NONAKTIF	NULL	NULL	NULL	NONAKTIF
NONAKTIF	NULL	NONAKTIF	NULL	NULL	NULL	NONAKTIF
NONAKTIF	NULL	NONAKTIF	NULL	NULL	NULL	NONAKTIF
NONAKTIF	NULL	NONAKTIF	NULL	NULL	NULL	NONAKTIF
NONAKTIF	NULL	NONAKTIF	NULL	NULL	NULL	NONAKTIF
NONAKTIF	NULL	NONAKTIF	NULL	NULL	NULL	NONAKTIF
NONAKTIF	NULL	NONAKTIF	NULL	NULL	NULL	NONAKTIF

id_paket_pilates	kuota_pilates	tgl_update_yoga	status_yoga	id_paket_yoga	kuota_yoga
NULL	NULL	NULL	NONAKTIF	NULL	NULL
NULL	NULL	NULL	NONAKTIF	NULL	NULL
NULL	NULL	NULL	NONAKTIF	NULL	NULL
NULL	NULL	NULL	NONAKTIF	NULL	NULL
NULL	NULL	NULL	NONAKTIF	NULL	NULL
NULL	NULL	NULL	NONAKTIF	NULL	NULL
NULL	NULL	NULL	NONAKTIF	NULL	NULL
NULL	NULL	NULL	NONAKTIF	NULL	NULL
NULL	NULL	NULL	NONAKTIF	NULL	NULL
NULL	NULL	NULL	NONAKTIF	NULL	NULL
NULL	NULL	NULL	NONAKTIF	NULL	NULL
NULL	NULL	NULL	NONAKTIF	NULL	NULL
NULL	NULL	NULL	NONAKTIF	NULL	NULL
NULL	NULL	NULL	NONAKTIF	NULL	NULL
NULL	NULL	NULL	NONAKTIF	NULL	NULL

---

16. Membuat tabel kelas

```sql
CREATE TABLE kelas (
    id_kelas INT PRIMARY KEY AUTO_INCREMENT,
    nama_kelas VARCHAR(100) NOT NULL,
    id_layanan INT NOT NULL,
    id_instruktur INT NOT NULL,
    tingkat ENUM('Beginner', 'Intermediate', 'Advanced') NOT NULL,
    tanggal_kelas DATE NOT NULL,
    jam_mulai TIME NOT NULL,
    jam_selesai TIME NOT NULL,
    kuota INT NOT NULL,
    FOREIGN KEY (id_layanan) REFERENCES layanan(id_layanan),
    FOREIGN KEY (id_instruktur) REFERENCES instruktur(id_instruktur)
);
```

Tabel kelas dalam basis data ini menampilkan jadwal kelas yang tersedia untuk layanan pilates dan yoga. Pada tabel ini, primary key-nya ‘id_kelas’ yang diatur sebagai auto increment, sedangkan ‘id_layanan’ adalah foreign key yang merujuk pada layanan yang menyediakan kelas tersebut dan ‘id_instruktur’ adalah foreign key yang merujuk pada instruktur yang mengajar pada kelas tersebut.

17. Menambahkan kolom pada tabel kelas

```sql
ALTER TABLE kelas
ADD durasi_kelas INT NOT NULL,
ADD sisa_kuota INT NOT NULL,
ADD status_kelas ENUM('TERSEDIA', 'KUOTA PENUH', 'SELESAI',
'DIBATALKAN') DEFAULT 'TERSEDIA';
```

Kolom ‘durasi_kelas’ menunjukkan lama waktu kelas berlangsung dalam menit, ‘sisa_kuota’ menunjukkan sisa kuota kelas tersedia yang dapat dipesan oleh member, dan ‘status_kelas’ menunjukkan apakah kelas tersebut ‘TERSEDIA’ (tersedia untuk dipesan), ‘KUOTA PENUH’ (kuota kelas sudah penuh), ‘SELESAI’ (kelas sudah selesai dilaksanakan), atau ‘DIBATALKAN’ (kelas dibatalkan oleh admin), dengan nilai default ‘TERSEDIA’.

18. Input nilai kolom pada tabel kelas secara otomatis

Kolom-kolom yang baru ditambahkan dengan perintah ALTER – ADD akan diatur untuk input otomatis menggunakan TRIGGER.

a. ‘durasi_kelas’

```
DELIMITER //
CREATE TRIGGER durasi_kelas
BEFORE INSERT ON kelas
FOR EACH ROW
BEGIN
    SET NEW.durasi_kelas = TIMESTAMPDIFF(MINUTE, NEW.jam_mulai,
    NEW.jam_selesai);
END;
//
DELIMITER ;
```

Nilai kolom ‘durasi_kelas’ akan dihitung sebagai selisih (dalam menit) dari jam mulai kelas dan jam selesai kelas dengan TIMESTAMPDIFF().

b. ‘sisa_kuota’

```
DELIMITER //
CREATE TRIGGER set_sisa_kuota
BEFORE INSERT ON kelas
FOR EACH ROW
BEGIN
    SET NEW.sisa_kuota = NEW.kuota;
END;
//
DELIMITER ;
```

Nilai kolom ‘sisa_kuota’ akan diatur sama dengan nilai ‘kuota’. Pada langkah selanjutnya, akan dibuat fungsi yang dapat mengurangi nilai dari ‘sisa_kuota’ apabila terdapat member yang mendaftar pada kelas tersebut.

19. Menambahkan data pada tabel kelas

Berikut adalah contoh data jadwal kelas yang tersedia,

```
INSERT INTO kelas (nama_kelas, id_layanan, id_instruktur, tingkat, tanggal_kelas, jam_mulai,
jam_selesai, kuota) VALUES
('Mat Pilates Intro',       2, 6, 'Beginner',     '2025-07-06', '08:00:00', '09:00:00', 12),
('Pilates Core Focus',      2, 7, 'Intermediate', '2025-07-07', '10:00:00', '11:30:00', 10),
('Reformer Strength',       2, 8, 'Advanced',     '2025-07-08', '17:00:00', '18:30:00', 8),
('Pilates Flow Balance',    2, 10, 'Beginner',     '2025-07-09', '09:00:00', '10:00:00', 15),
('Hatha Yoga Morning',      3, 12, 'Beginner',     '2025-07-06', '07:30:00', '08:30:00', 15),
('Vinyasa Flow Intermediate',3, 14, 'Intermediate','2025-07-08', '16:00:00', '17:00:00', 12),
('Yin Yoga Deep Stretch',   3, 15, 'Advanced',     '2025-07-10', '18:00:00', '19:30:00', 10);
```

20. Menampilkan isi tabel kelas

```
SELECT * FROM kelas;
```

Maka, tabel yang akan ditampilkan adalah,

id_kelas	nama_kelas	id_layanan	id_instruktur	tingkat	tanggal_kelas	jam_mulai	jam_selesai
1	Mat Pilates Intro	2	6	Beginner	2025-07-06	08:00:00	09:00:00
2	Pilates Core Focus	2	7	Intermediate	2025-07-07	10:00:00	11:30:00
3	Reformer Strength	2	8	Advanced	2025-07-08	17:00:00	18:30:00
4	Pilates Flow Balance	2	10	Beginner	2025-07-09	09:00:00	10:00:00
5	Hatha Yoga Morning	3	12	Beginner	2025-07-06	07:30:00	08:30:00
6	Vinyasa Flow Intermediate	3	14	Intermediate	2025-07-08	16:00:00	17:00:00
7	Yin Yoga Deep Stretch	3	15	Advanced	2025-07-10	18:00:00	19:30:00

kuota	durasi_kelas	sisa_kuota	status_kelas
12	60	12	TERSEDIA
10	90	10	TERSEDIA
8	90	8	TERSEDIA
15	60	15	TERSEDIA
15	60	15	TERSEDIA
12	60	12	TERSEDIA
10	90	10	TERSEDIA

---

21. Membuat tabel transaksi

```
CREATE TABLE transaksi (
    id_transaksi INT PRIMARY KEY AUTO_INCREMENT,
    id_member INT NOT NULL,
    id_paket INT NOT NULL,
    tanggal_transaksi DATE NOT NULL,
    FOREIGN KEY (id_member) REFERENCES member(id_member),
    FOREIGN KEY (id_paket) REFERENCES paket(id_paket)
);
```

Tabel transaksi dalam basis data ini menyimpan daftar transaksi dari pembelian paket yang dilakukan oleh member. Pada tabel ini, primary key-nya adalah ‘id_transaksi’ yang diatur sebagai auto increment, sedangkan ‘id_member’ adalah foreign key yang merujuk pada member yang melakukan transaksi dan ‘id_paket’ adalah foreign key yang merujuk pada paket yang dibeli oleh member.

22. Menambahkan kolom pada tabel transaksi

```
ALTER TABLE transaksi
ADD id_layanan INT NOT NULL,
ADD jumlah_pembayaran DECIMAL(12,2) NOT NULL,
ADD CONSTRAINT fk_layanan FOREIGN KEY (id_layanan) REFERENCES layanan(id_layanan);
```

Kolom ‘id_layanan’ adalah foreign key yang merujuk pada layanan yang menyediakan paket tersebut dan kolom ‘jumlah_pembayaran’ menunjukkan total harga yang perlu dibayarkan oleh member ketika melakukan pembelian paket.

23. Input nilai kolom pada tabel transaksi secara otomatis
    a. ‘id_layanan’

```
DELIMITER //
CREATE TRIGGER set_id_layanan
BEFORE INSERT ON transaksi
FOR EACH ROW
BEGIN
    DECLARE layanan INT;

    SELECT id_layanan INTO layanan
    FROM paket
    WHERE id_paket = NEW.id_paket
    LIMIT 1;

    SET NEW.id_layanan = layanan;
END;
//
DELIMITER ;
```

Melalui foreign key ‘id_paket’, TRIGGER akan menginput ‘id_layanan’ secara otomatis dengan mengambil informasi dari tabel paket.

b. ‘jumlah_pembayaran

```
DELIMITER //
CREATE TRIGGER set_jumlah_pembayaran
BEFORE INSERT ON transaksi
FOR EACH ROW
BEGIN
    DECLARE jumlah_pembayaran DECIMAL(12,2);

    SELECT harga INTO jumlah_pembayaran
    FROM paket
    WHERE id_paket = NEW.id_paket;

    SET NEW.jumlah_pembayaran = jumlah_pembayaran;
END;
//
DELIMITER ;
```

Melalui foreign key ‘id_paket’, TRIGGER akan menginput ‘jumlah_pembayaran’ secara otomatis dengan mengambil informasi ‘harga’ dari tabel paket.

24. Menambahkan data pada tabel transaksi
    Berikut adalah contoh daftar transaksi,

```sql
INSERT INTO transaksi (id_member, id_paket, tanggal_transaksi)
VALUES
(1, 1, '2025-07-01'),
(2, 2, '2025-07-01'),
(3, 3, '2025-07-02'),
(4, 4, '2025-07-02'),
(5, 5, '2025-07-03'),
(6, 6, '2025-07-03'),
(1, 7, '2025-07-04'),
(2, 8, '2025-07-04'),
(3, 9, '2025-07-05'),
(4, 1, '2025-07-05'),
(1, 2, '2025-07-06'),
(5, 3, '2025-07-06'),
(6, 4, '2025-07-07'),
(3, 5, '2025-07-07'),
(4, 6, '2025-07-08'),
(2, 7, '2025-07-08'),
(1, 8, '2025-07-09'),
(2, 9, '2025-07-09'),
(3, 1, '2025-07-10'),
(4, 2, '2025-07-10');
```

25. Menampilkan isi tabel transaksi

```sql
SELECT * FROM transaksi;
```

Maka, tabel yang akan ditampilkan adalah,

id_transaksi	id_member	id_paket	tanggal_transaksi	id_layanan	jumlah_pembayaran
1	1	1	2025-07-01	1	750000.00
2	2	2	2025-07-01	1	2000000.00
3	3	3	2025-07-02	1	7500000.00
4	4	4	2025-07-02	1	100000.00
5	5	5	2025-07-03	2	850000.00
6	6	6	2025-07-03	2	1600000.00
7	1	7	2025-07-04	2	2300000.00
8	2	8	2025-07-04	2	200000.00
9	3	9	2025-07-05	3	700000.00
10	4	1	2025-07-05	1	750000.00
11	1	2	2025-07-06	1	2000000.00
12	5	3	2025-07-06	1	7500000.00
13	6	4	2025-07-07	1	100000.00
14	3	5	2025-07-07	2	850000.00
15	4	6	2025-07-08	2	1600000.00

---

26. Membuat tabel booking

```
CREATE TABLE booking (
    id_booking INT PRIMARY KEY AUTO_INCREMENT,
    id_member INT NOT NULL,
    id_kelas INT NOT NULL,
    tanggal_booking DATE NOT NULL,
    FOREIGN KEY (id_member) REFERENCES member(id_member),
    FOREIGN KEY (id_kelas) REFERENCES kelas(id_kelas)
);
```

Tabel booking dalam basis data ini menyimpan informasi mengenai daftar pemesanan kelas yang dilakukan oleh member. Pada tabel ini, primary key-nya adalah 'id_booking' yang diatur sebagai auto increment, sedangkan 'id_member' adalah foreign key yang merujuk pada member yang melakukan pemesanan, dan 'id_kelas' adalah foreign key yang merujuk pada kelas yang dipesan oleh member.

27. Menambahkan kolom pada tabel booking

```
ALTER TABLE booking
ADD id_layanan INT NOT NULL,
ADD CONSTRAINT fk2_layanan FOREIGN KEY (id_layanan) REFERENCES layanan(id_layanan);
```

Kolom 'id_layanan' adalah foreign key yang merujuk pada layanan yang menyediakan kelas yang dipesan oleh member.

28. Input nilai kolom pada tabel booking secara otomatis

```
DELIMITER //
CREATE TRIGGER set_id_layanan2
BEFORE INSERT ON booking
FOR EACH ROW
BEGIN
    DECLARE layanan INT;

    SELECT id_layanan INTO layanan
    FROM kelas
    WHERE id_kelas = NEW.id_kelas
    LIMIT 1;

    SET NEW.id_layanan = layanan;
END;
//
DELIMITER ;
```

Melalui foreign key 'id_kelas', TRIGGER akan menginput 'id_layanan' secara otomatis dengan mengambil informasi dari tabel kelas.

29. Menambahkan data pada tabel booking

Berikut adalah contoh daftar pemesanan kelas,

```
INSERT INTO booking(id_member, id_kelas, tanggal_booking) VALUES
(1, 1, '2025-07-01'),
(2, 2, '2025-07-01'),
(3, 3, '2025-07-02'),
(4, 4, '2025-07-02'),
(5, 5, '2025-07-03'),
(6, 6, '2025-07-03'),
(1, 7, '2025-07-04'),
(2, 3, '2025-07-04'),
(3, 5, '2025-07-05'),
(4, 2, '2025-07-05'),
(1, 2, '2025-07-06'),
(5, 3, '2025-07-06'),
(6, 4, '2025-07-07'),
(3, 5, '2025-07-07'),
(4, 6, '2025-07-08'),
(2, 7, '2025-07-08'),
(1, 4, '2025-07-09'),
(2, 3, '2025-07-09'),
(3, 1, '2025-07-10'),
(4, 1, '2025-07-10');
```

30. Menampilkan isi tabel booking

```
SELECT * FROM booking;
```

Maka, tabel yang akan ditampilkan adalah,

	id_booking	id_member	id_kelas	tanggal_booking	id_layanan
	1	1	1	2025-07-01	2
	2	2	2	2025-07-01	2
	3	3	3	2025-07-02	2
	4	4	4	2025-07-02	2
	5	5	5	2025-07-03	3
	6	6	6	2025-07-03	3
	7	1	7	2025-07-04	3
	8	2	3	2025-07-04	2
	9	3	5	2025-07-05	3
	10	4	2	2025-07-05	2
	11	1	2	2025-07-06	2
	12	5	3	2025-07-06	2
	13	6	4	2025-07-07	2
	14	3	5	2025-07-07	3
	15	4	6	2025-07-08	3

---

31. Memperbarui informasi paket aktif pada tabel member

Pada langkah pertama, kita akan membuat fungsi STORED PROCEDURE yang dinamakan dengan refresh_member_data().

```
DELIMITER //
CREATE PROCEDURE refresh_member_data()
BEGIN
```

Kemudian, berdasarkan tabel transaksi, untuk setiap member, diambil tanggal transaksi terbaru. Jika paket yang dibeli merupakan paket dari layanan gym,

```
    UPDATE member m
    JOIN (
        SELECT id_member, id_paket, MAX(tanggal_transaksi) AS tanggal_terbaru
        FROM transaksi
        WHERE id_layanan = 1
        GROUP BY id_member, id_paket
    ) b ON m.id_member = b.id_member
    JOIN paket pm ON b.id_paket = pm.id_paket
    SET
        m.id_paket_gym = b.id_paket,
        m.tgl_update_gym = b.tanggal_terbaru,
        m.kuota_gym = pm.durasi,
        m.status_gym = CASE
            WHEN pm.durasi > 0 THEN 'AKTIF'
            ELSE 'NONAKTIF'
        END;
```

Jika paket yang dibeli merupakan paket dari layanan pilates,

```
    UPDATE member m
    JOIN (
        SELECT id_member, id_paket, MAX(tanggal_transaksi) AS tanggal_terbaru
        FROM transaksi
        WHERE id_layanan = 2
        GROUP BY id_member, id_paket
    ) b ON m.id_member = b.id_member
    JOIN paket pm ON b.id_paket = pm.id_paket
    SET
        m.id_paket_pilates = b.id_paket,
        m.tgl_update_pilates = b.tanggal_terbaru,
        m.kuota_pilates = pm.durasi,
        m.status_pilates = CASE
            WHEN pm.durasi > 0 THEN 'AKTIF'
            ELSE 'NONAKTIF'
        END;
```

Jika paket yang dibeli merupakan paket dari layanan yoga,

```sql
UPDATE member m
JOIN (
    SELECT id_member, id_paket, MAX(tanggal_transaksi) AS tanggal_terbaru
    FROM transaksi
    WHERE id_layanan = 3
    GROUP BY id_member, id_paket
) b ON m.id_member = b.id_member
JOIN paket pm ON b.id_paket = pm.id_paket
SET
    m.id_paket_yoga = b.id_paket,
    m.tgl_update_yoga  = b.tanggal_terbaru,
    m.kuota_yoga  = pm.durasi,
    m.status_yoga  = CASE
        WHEN pm.durasi > 0 THEN 'AKTIF'
        ELSE 'NONAKTIF'
    END;
```

Kolom ‘tgl_update_gym’, ‘status_gym’, ‘id_paket_gym’, dan ‘kuota_gym’ akan terinput secara otomatis apabila member setidaknya pernah melakukan transaksi untuk layanan gym, begitupun untuk layanan pilates dan yoga. Pada saat member melalukan transaksi pembelian paket, kuota akan diinput sesuai dengan kuota yang didapatkan sesuai dengan paket yang dibeli, maka status untuk layanan tersebut akan menjadi aktif (nilai kuota lebih dari nol).

Selanjutnya adalah pengurangan kuota seiring dengan penggunaan layanan oleh member. Apabila member memiliki paket layanan gym yang aktif,

```sql
UPDATE member m
JOIN (
    SELECT m.id_member, p.durasi, m.tgl_update_gym
    FROM member m
    JOIN paket p ON m.id_paket_gym = p.id_paket
    WHERE p.id_layanan = 1
) AS sub ON m.id_member = sub.id_member
SET
    m.kuota_gym = GREATEST(0, sub.durasi - DATEDIFF(CURRENT_DATE(), sub.tgl_update_gym)),
    m.status_gym = CASE
        WHEN (sub.durasi - DATEDIFF(CURRENT_DATE(), sub.tgl_update_gym)) <= 0 THEN 'NONAKTIF'
        ELSE m.status_gym
    END;
```

Jumlah kuota yang didapatkan oleh member sesuai dengan paket yang dibeli akan dikurangi dengan selisih tanggal hari ini (CURRENT_DATE()) dengan tanggal pembelian paket. Apabila kuota sama dengan nol, maka ‘status_paket’ akan berubah menjadi ‘NONAKTIF’.

Apabila member memiliki paket layanan pilates dan yoga yang aktif,

```
UPDATE member m
JOIN (
    SELECT bk.id_member, COUNT(*) AS total_pilates
    FROM booking bk
    JOIN kelas k ON bk.id_kelas = k.id_kelas
    WHERE k.id_layanan = 2
    GROUP BY bk.id_member
) AS sub ON m.id_member = sub.id_member
SET
    m.kuota_pilates = GREATEST(0, m.kuota_pilates - sub.total_pilates),
    m.status_pilates = CASE
        WHEN (m.kuota_pilates - sub.total_pilates) <= 0 THEN 'NONAKTIF'
        ELSE m.status_pilates
    END;
```

```
UPDATE member m
JOIN (
    SELECT bk.id_member, COUNT(*) AS total_yoga
    FROM booking bk
    JOIN kelas k ON bk.id_kelas = k.id_kelas
    WHERE k.id_layanan = 3
    GROUP BY bk.id_member
) AS sub ON m.id_member = sub.id_member
SET
    m.kuota_yoga = GREATEST(0, m.kuota_yoga- sub.total_yoga),
    m.status_yoga = CASE
        WHEN (m.kuota_yoga - sub.total_yoga) <= 0 THEN 'NONAKTIF'
        ELSE m.status_yoga
    END;
```

Jumlah kuota yang didapatkan oleh member sesuai dengan paket yang dibeli akan dikurangi dengan jumlah kelas yang dipesan oleh member berdasarkan ‘id_layanan’. Apabila kuota sama dengan nol, maka ‘status_paket’ akan berubah menjadi ‘NONAKTIF’.

```
CREATE TEMPORARY TABLE IF NOT EXISTS temp_member_aktif AS
SELECT id_member
FROM member
WHERE status_gym = 'AKTIF'
    OR status_pilates = 'AKTIF'
    OR status_yoga = 'AKTIF';

UPDATE member
JOIN temp_member_aktif t ON member.id_member = t.id_member
SET member.status_member = 'AKTIF';
```

Apabila member memiliki setidaknya satu paket layanan yang aktif, maka ‘status_member’ adalah ‘AKTIF’.

Setelah pembaruan pada tanggal, status, jenis paket, dan sisa kuota, akan ditampilkan tabel member yang terbaru,

```
    SELECT * FROM member;
END;
//
DELIMITER ;
```

Untuk memanggil fungsi, gunakan perintah,

```
CALL refresh_member_data();
```

Maka, fungsi akan menampilkan tabel member yang terbaru,

id_member	nama	email	no_hp	tanggal_daftar	status_member	tgl_update_gym	status_gym	id_paket_gym	kuota_gym
1	Andi Saputra	andi.s@example.com	081234567801	2025-07-01	AKTIF	2025-07-01	AKTIF	1	24
2	Bella Rahma	bella.r@example.com	081234567802	2025-07-01	AKTIF	2025-07-01	AKTIF	2	84
3	Citra Dewi	citra.d@example.com	081234567803	2025-07-01	AKTIF	2025-07-02	AKTIF	3	355
4	Dedi Pratama	dedi.p@example.com	081234567804	2025-07-02	AKTIF	2025-07-02	NONAKTIF	4	0
5	Eka Wulandari	eka.w@example.com	081234567805	2025-07-02	AKTIF	2025-07-06	AKTIF	3	359

tgl_update_pilates	status_pilates	id_paket_pilates	kuota_pilates	tgl_update_yoga	status_yoga	id_paket_yoga	kuota_yoga
2025-07-04	AKTIF	7	9	NULL	NONAKTIF	NULL	NULL
2025-07-04	NONAKTIF	8	0	2025-07-09	AKTIF	9	3
2025-07-07	AKTIF	5	2	2025-07-05	AKTIF	9	2

32. Memperbarui kuota dan kelas pada tabel kelas

Pada langkah pertama, kita akan membuat fungsi STORED PROCEDURE yang dinamakan dengan refresh_kelas().

```
DELIMITER //
CREATE PROCEDURE refresh_kelas()
BEGIN
```

Selanjutnya, untuk setiap baris data untuk pemesanan suatu kelas pada tabel booking, kuota kelas akan berkurang sebanyak satu kuota.

```
    DROP TEMPORARY TABLE IF EXISTS temp_kuota;
    CREATE TEMPORARY TABLE temp_kuota AS
    SELECT
        k.id_kelas,
        GREATEST(k.sisa_kuota - IFNULL(b.jumlah_booking, 0), 0) AS kuota_baru
    FROM kelas k
    LEFT JOIN (
        SELECT id_kelas, COUNT(*) AS jumlah_booking
        FROM booking
        GROUP BY id_kelas
    ) b ON k.id_kelas = b.id_kelas;

    UPDATE kelas k
    JOIN temp_kuota t ON k.id_kelas = t.id_kelas
    SET k.sisa_kuota = t.kuota_baru;
```

Dengan nilai default ‘TERSEDIA’, kelas masih dapat dipesan. Namun, apabila kuota kelas sama dengan nol, maka status kelas akan berubah menjadi ‘KUOTA HABIS’. Apabila tanggal hari ini sudah melewati tanggal kelas dilaksanakan, maka status kelas akan berubah menjadi ‘SELESAI’. Status kelas ‘DIBATALKAN’ dapat diinput secara manual oleh admin.

```
    DROP TEMPORARY TABLE IF EXISTS temp_status;
    CREATE TEMPORARY TABLE temp_status AS
    SELECT
        k.id_kelas,
        CASE
            WHEN k.tanggal_kelas < CURRENT_DATE() THEN 'SELESAI'
            WHEN k.sisa_kuota = 0 THEN 'KUOTA HABIS'
            ELSE 'TERSEDIA'
        END AS status_baru
    FROM kelas k
    WHERE k.status_kelas != 'DIBATALKAN';

    UPDATE kelas k
    JOIN temp_status s ON k.id_kelas = s.id_kelas
    SET k.status_kelas = s.status_baru;
```

Setelah pembaruan pada kuota dan status, akan ditampilkan tabel kelas yang terbaru,

```
    SELECT * FROM kelas;
END;
//
DELIMITER ;
```

Untuk memanggil fungsi, gunakan perintah,

```
CALL refresh_kelas();
```

Maka, fungsi akan menampilkan tabel kelas yang terbaru,

id_kelas	nama_kelas	id_layanan	id_instruktur	tingkat	tanggal_kelas	jam_mulai	jam_selesai	kuota	durasi_kelas	sisa_kuota	status_kelas
1	Mat Pilates Intro	2	6	Beginner	2025-07-06	08:00:00	09:00:00	12	60	9	SELESAI
2	Pilates Core Focus	2	7	Intermediate	2025-07-07	10:00:00	11:30:00	10	90	7	TERSEDIA
3	Reformer Strength	2	8	Advanced	2025-07-08	17:00:00	18:30:00	8	90	4	TERSEDIA
4	Pilates Flow Balance	2	10	Beginner	2025-07-09	09:00:00	10:00:00	15	60	12	TERSEDIA
5	Hatha Yoga Morning	3	12	Beginner	2025-07-06	07:30:00	08:30:00	15	60	12	SELESAI
6	Vinyasa Flow Intermediate	3	14	Intermediate	2025-07-08	16:00:00	17:00:00	12	60	10	TERSEDIA
7	Yin Yoga Deep Stretch	3	15	Advanced	2025-07-10	18:00:00	19:30:00	10	90	8	TERSEDIA

Setelah merancang struktur basis data berupa tabel dan menambahkan data tabel. Kita akan melakukan analisis dengan perintah agregasi pada tabel transaksi.

Catatan: Query agregasi di bawah ini hanya diterapkan sebagai contoh pada tabel transaksi dan dapat diterapkan pada tabel lainnya sesuai kebutuhan analisis.

33. Menampilkan pemasukkan berdasarkan tanggal

```
SELECT
    tanggal_transaksi,
    SUM(jumlah_pembayaran) AS total_pembayaran
FROM transaksi
GROUP BY tanggal_transaksi
ORDER BY tanggal_transaksi;
```

Maka, hasil yang akan keluar adalah

	tanggal_transaksi	total_pembayaran
▶	2025-07-01	2750000.00
	2025-07-02	7600000.00
	2025-07-03	2450000.00
	2025-07-04	2500000.00
	2025-07-05	1450000.00
	2025-07-06	9500000.00
	2025-07-07	950000.00
	2025-07-08	3900000.00
	2025-07-09	900000.00
	2025-07-10	2750000.00

Apabila tanggal transaksi sudah melewati hitungan hari (bulan atau tahun), perintah untuk menjumlahkan pembayaran berdasarkan bulan dapat dimodifikasi sebagai berikut,

```
SELECT
    MONTH(tanggal_transaksi),
    SUM(jumlah_pembayaran) AS total_pembayaran
FROM transaksi
GROUP BY MONTH(tanggal_transaksi)
ORDER BY MONTH(tanggal_transaksi);
```

atau

```
SELECT
    YEAR(tanggal_transaksi),
    SUM(jumlah_pembayaran) AS total_pembayaran
FROM transaksi
GROUP BY YEAR(tanggal_transaksi)
ORDER BY YEAR(tanggal_transaksi);
```

34. Menampilkan tanggal dengan pemasukkan terendah atau tertinggi

    Untuk menampilkan pemasukkan terendah, berarti kita menampilkan total jumlah pembayaran minimum,

```
SELECT
    tanggal_transaksi,
    total_pembayaran
FROM (
    SELECT
        tanggal_transaksi,
        SUM(jumlah_pembayaran) AS total_pembayaran
    FROM transaksi
    GROUP BY tanggal_transaksi
) AS hasil_sum
WHERE total_pembayaran = (
    SELECT MIN(total_pembayaran)
    FROM (
        SELECT
            tanggal_transaksi,
            SUM(jumlah_pembayaran) AS total_pembayaran
        FROM transaksi
        GROUP BY tanggal_transaksi
    ) AS sub_min
);
```

Untuk menampilkan pemasukkan terendah, berarti kita menampilkan total jumlah pembayaran maksimum,

```
SELECT
    tanggal_transaksi,
    total_pembayaran
FROM (
    SELECT
        tanggal_transaksi,
        SUM(jumlah_pembayaran) AS total_pembayaran
    FROM transaksi
    GROUP BY tanggal_transaksi
) AS hasil_sum
WHERE total_pembayaran = (
    SELECT MAX(total_pembayaran)
    FROM (
        SELECT
            tanggal_transaksi,
            SUM(jumlah_pembayaran) AS total_pembayaran
        FROM transaksi
        GROUP BY tanggal_transaksi
    ) AS sub_min
);
```

Perintah di atas melakukan SUM terlebih dahulu untuk menjumlahkan ‘jumlah_pembayaran’ berdasarkan tanggal kemudian mengambil tanggal dengan nilai minimum atau maksimum.

Apabila owner/user ingin menghapus data dari basis data, maka jalanan perintah,

35. Menghapus basis data

```
DROP DATABASE studio;
```

Apabila perintah ini dijalankan, seluruh struktur dan isi dalam basis data akan terhapus.

Selanjutnya, contoh perintah akan diterapkan pada tabel booking,

36. Menghapus tabel dalam basis data

```
DROP TABLE booking;
```

Maka, struktur dan isi tabel akan terhapus.

37. Menghapus seluruh data dalam suatu tabel

```
TRUNCATE booking;
```

Maka, seluruh data dalam tabel akan terhapus, tetapi struktur dari tabel tersebut masih ada.

38. Menghapus suatu kolom dalam suatu tabel

```
ALTER TABLE booking
DROP COLUMN tanggal_booking;
```

Maka, kolom ‘tanggal_booking’ akan terhapus dari tabel.

39. Menghapus suatu baris data dalam suatu tabel

```
DELETE FROM booking
WHERE id_booking = 1;
```

Maka, baris data dengan id_booking = 1 akan terhapus dari tabel.

Untuk implementasi di dunia nyata, pemilik basis data dapat memperketat akses untuk pengguna berupa pembatasan terhadap perintah penghapusan data seperti DROP. Hal ini merupakan bentuk antisipasi untuk mencegah kehilangan data penting secara tidak sengaja maupun karena tindakan yang disengaja (malicious). Dengan membatasi hak akses, hanya administrator atau pengguna tertentu yang memiliki wewenang penuh terhadap struktur basis data, sehingga integritas dan keamanan sistem dapat lebih terjaga.

Selain itu, basis data juga dapat diintegrasikan dengan bahasa pemrograman lainnya untuk keperluan pengembangan aplikasi berbasis web maupun desktop. Melalui integrasi ini, aplikasi dapat mengambil, menampilkan, dan memproses data dari database tanpa harus memberikan akses penuh kepada pengguna, sehingga lebih aman dan fleksibel.