

**LAPORAN RESMI**  
**FINAL PROJECT SISTEM BASIS DATA**



Kelompok 2

Marcelinus Alvinanda Chrisantya / 5027221012

Etha Felisya Br Purba / 5027221017

Salsabila Amalia Harjanto / 5027221023

Dosen Pengampu: Hafara Firdausi, S.Kom., M.Kom.

**INSTITUT TEKNOLOGI SEPULUH NOPEMBER**  
**SURABAYA**  
**2022/2023**

# **BAB I**

## **DESKRIPSI SKENARIO**

### **1.1 Nama Sistem Manajemen**

MedCare Management System.

### **1.2 Deskripsi Sistem**

MedCare Management System merupakan sistem manajemen terintegrasi yang dirancang khusus untuk meningkatkan efisiensi dan kualitas layanan di rumah sakit dan lembaga kesehatan. Salah satu fitur utamanya adalah kemampuannya dalam manajemen pasien. Dengan sistem ini, petugas administrasi dapat dengan mudah mendaftarkan pasien baru dengan informasi pribadi, riwayat penyakit, dan detail asuransi. Selain itu, sistem ini memfasilitasi pembaruan informasi pasien secara berkala, memastikan bahwa rekam medis selalu terkini dan akurat.

Rekam Medis Elektronik menjadi tulang punggung operasional rumah sakit melalui sistem ini. Dokter dan perawat dapat mengakses informasi medis pasien sesuai dengan peran masing-masing. Hal ini memungkinkan pemberian pelayanan kesehatan yang lebih personal dan terfokus pada kebutuhan individual pasien. Selain itu, MedCare Management System memberikan dukungan penuh dalam pengelolaan appointment, mencakup rawat inap, rawat jalan, dan prosedur medis, yang memberikan keberlanjutan pada proses pelayanan.

Fungsi manajemen inventaris diimplementasikan dengan baik dalam sistem ini. Petugas dapat dengan mudah memantau stok obat, alat medis, dan peralatan rumah sakit. Proses pemantauan stok dilakukan untuk mengoptimalkan efisiensi pengelolaan inventaris dan memastikan kelancaran operasional sehari-hari.

Jadwal dokter dan perawat dikelola secara efisien dan terstruktur melalui sistem ini. Dari penjadwalan hingga pembaruan, MedCare Management System menyediakan informasi kontak dan spesialisasi masing-masing tenaga medis, memastikan penempatan yang optimal sesuai dengan kebutuhan pelayanan kesehatan.

Dalam aspek administrasi dan pembayaran, sistem ini memberikan solusi terpadu. Tagihan dibuat secara otomatis berdasarkan layanan yang diberikan, dan data asuransi pasien dikelola dengan baik. Petugas administrasi mencatat penerimaan dan pembayaran, menjaga kelancaran administrasi keuangan rumah sakit.

Di bidang manajemen laboratorium, MedCare Management System mencatat dan melacak hasil tes laboratorium. Informasi ini diintegrasikan dengan rekam medis untuk memberikan pandangan yang lebih komprehensif tentang kondisi pasien. Keseluruhan, sistem ini bukan hanya alat manajemen, melainkan mitra yang mendukung kolaborasi medis dan meningkatkan kualitas pelayanan kesehatan di setiap tingkatan.

## 2.1 Visualisasi ERD

[illegible]

## **BAB III**

### **DESKRIPSI TABEL DAN KOLOM**

#### **3.1 Data yang Akan Disimpan**

##### **1. Tabel pasien**

Deskripsi: Tabel ini digunakan untuk menyimpan informasi tentang pasien dalam sebuah sistem kesehatan atau rumah sakit.

Kolom:

- a. ID Pasien (id\_pasien): Tipe data: Integer. Deskripsi: Kunci utama yang secara unik mengidentifikasi setiap pasien.
- b. Nama Pasien (nama\_pasien): Tipe data: Varchar(100). Deskripsi: Menyimpan nama lengkap pasien.
- c. Tanggal Lahir (tanggal\_lahir): Tipe data: Date. Deskripsi: Menyimpan tanggal lahir pasien.
- d. Jenis Kelamin (jenis\_kelamin): Tipe data: Enum ('Laki-laki', 'Perempuan'). Deskripsi: Menyimpan informasi jenis kelamin pasien.

##### **2. Tabel asuransi**

Deskripsi: Tabel ini digunakan untuk menyimpan informasi tentang asuransi yang dimiliki oleh pasien.

Kolom:

- a. Nomor Asuransi (nomor\_asuransi): Tipe data: Varchar(20). Deskripsi: Kunci utama yang secara unik mengidentifikasi setiap nomor asuransi.
- b. Nama Perusahaan Asuransi (nama\_perusahaan\_asuransi): Tipe data: Varchar(100). Deskripsi: Menyimpan nama perusahaan asuransi yang menyediakan layanan kepada pasien.
- c. ID Pasien (id\_pasien): Tipe data: Integer. Deskripsi: Kunci asing (FK) yang mengacu pada kolom "id\_pasien" dalam tabel "Pasien". Menyediakan hubungan antara data asuransi dan pasien yang terkait.

##### **3. Tabel asuransi\_pasien**

Deskripsi: Tabel ini digunakan untuk menghubungkan informasi antara pasien dan nomor asuransi yang terkait dengan pasien tersebut.

Kolom:

- a. ID Pasien (id\_pasien): Tipe data: Integer. Deskripsi: Kunci utama yang merupakan kunci asing (FK) yang mengacu pada kolom "id\_pasien" dalam tabel "Pasien". Menyediakan hubungan antara data asuransi dan data pasien.
- b. Nomor Asuransi (nomor\_asuransi): Tipe data: Varchar(50). Deskripsi: Kunci asing (FK) yang mengacu pada nomor asuransi dalam tabel lain yang menyimpan informasi asuransi.

#### 4. Tabel tenaga\_medis

Deskripsi: Tabel ini digunakan untuk menyimpan informasi tentang tenaga medis dalam suatu lembaga atau rumah sakit.

Kolom:

- a. ID Tenaga Medis (id\_tenaga\_medis): Tipe data: Integer. Deskripsi: Kunci utama yang secara unik mengidentifikasi setiap tenaga medis.
- b. Nama Tenaga Medis (nama\_tenaga\_medis): Tipe data: Varchar(100). Deskripsi: Menyimpan nama lengkap tenaga medis.

#### 5. Tabel spesialisasi\_tenaga\_medis

Deskripsi: Tabel ini digunakan untuk menyimpan informasi tentang spesialisasi dan jenis pekerjaan yang dimiliki oleh tenaga medis.

Kolom:

- a. ID Spesialisasi (id\_spesialisasi): Tipe data: Integer. Deskripsi: Kunci utama yang secara unik mengidentifikasi setiap entri spesialisasi.
- b. ID Tenaga Medis (id\_tenaga\_medis): Tipe data: Integer. Deskripsi: Kunci asing (FK) yang mengacu pada kolom "id\_tenaga\_medis" dalam tabel "Tenaga\_Medis". Menyediakan hubungan antara data tenaga medis dan spesialisasi.
- c. Jenis Pekerjaan (jenis\_pekerjaan): Tipe data: Varchar(50). Deskripsi: Menyimpan informasi tentang jenis pekerjaan atau peran spesifik tenaga medis.
- d. Spesialisasi Tenaga Medis (spesialisasi\_tenaga\_medis): Tipe data: Varchar(100). Deskripsi: Menyimpan informasi tentang spesialisasi keterampilan atau keahlian khusus yang dimiliki oleh tenaga medis.

#### 6. Tabel kontak\_dan\_gaji

Deskripsi: Tabel ini digunakan untuk menyimpan informasi kontak dan gaji tenaga medis.

Kolom:

- a. Kontak Tenaga Medis (kontak\_tenaga\_medis): Tipe data: Varchar(20). Deskripsi: Kunci utama yang secara unik mengidentifikasi setiap entri kontak tenaga medis.
- b. ID Tenaga Medis (id\_tenaga\_medis): Tipe data: Integer. Deskripsi: Kunci asing (FK) yang mengacu pada kolom "id\_tenaga\_medis" dalam tabel "Tenaga\_Medis". Menyediakan hubungan antara data kontak dan gaji dengan tenaga medis.
- c. Gaji Pokok (gaji\_pokok): Tipe data: Decimal(10,2). Deskripsi: Menyimpan informasi tentang gaji pokok tenaga medis.
- d. Gaji Bonus (gaji\_bonus): Tipe data: Decimal(10,2). Deskripsi: Menyimpan informasi tentang gaji bonus yang mungkin diterima tenaga medis.

#### 7. Tabel catatan\_pasien

Deskripsi: Tabel ini digunakan untuk menyimpan catatan medis pasien yang dihasilkan oleh tenaga medis.

Kolom:

- a. ID Rekam Medis (id\_rekam\_medis): Tipe data: Integer. Deskripsi: Kunci utama yang secara unik mengidentifikasi setiap rekam medis.
- b. ID Pasien (id\_pasien): Tipe data: Integer. Deskripsi: Kunci asing (FK) yang mengacu pada kolom "id\_pasien" dalam tabel "Pasien". Menyediakan hubungan antara data rekam medis dengan pasien.
- c. ID Tenaga Medis (id\_tenaga\_medis): Tipe data: Integer. Deskripsi: Kunci asing (FK) yang mengacu pada kolom "id\_tenaga\_medis" dalam tabel "Tenaga\_Medis". Menyediakan hubungan antara data rekam medis dengan tenaga medis yang merawat.
- d. Tanggal Catatan Medis (tanggal\_catatan\_medis): Tipe data: Date. Deskripsi: Menyimpan tanggal pembuatan catatan medis.

#### 8. Tabel diagnosa

Deskripsi: Tabel ini digunakan untuk menyimpan informasi diagnosa pasien.

Kolom:

- a. ID Diagnosa (id\_diagnosa): Tipe data: Integer. Deskripsi: Kunci utama yang secara unik mengidentifikasi setiap diagnosa.
- b. ID Rekam Medis (id\_rekam\_medis): Tipe data: Integer. Deskripsi: Kunci asing (FK) yang mengacu pada kolom "id\_rekam\_medis" dalam tabel "Catatan\_Pasien". Menyediakan hubungan antara data diagnosa dengan rekam medis.
- c. Diagnosa (diagnosa): Tipe data: Varchar(100). Deskripsi: Menyimpan informasi tentang diagnosa yang diberikan.
- d. Deskripsi Diagnosa (deskripsi\_diagnosa): Tipe data: Text. Deskripsi: Menyimpan deskripsi lebih lanjut tentang diagnosa.
- e. Metode Pengobatan (metode\_pengobatan): Tipe data: Text. Deskripsi: Menyimpan informasi tentang metode pengobatan yang direkomendasikan.

#### 9. Tabel inventaris

Deskripsi: Tabel ini digunakan untuk menyimpan informasi tentang inventaris barang.

Kolom:

- a. ID Inventaris (id\_inventaris): Tipe data: Integer. Deskripsi: Kunci utama yang secara unik mengidentifikasi setiap inventaris barang.
- b. Nama Barang (nama\_barang): Tipe data: Varchar(100). Deskripsi: Menyimpan nama barang dalam inventaris.
- c. Jumlah Stok Barang (jumlah\_stok\_barang): Tipe data: Integer. Deskripsi: Menyimpan jumlah stok barang yang tersedia.
- d. Lokasi Penyimpanan Barang (lokasi\_penyimpanan\_barang): Tipe data: Varchar(50). Deskripsi: Menyimpan informasi tentang lokasi penyimpanan barang.

#### 10. Tabel detail\_barang

Deskripsi: Tabel ini digunakan untuk menyimpan detail tambahan tentang setiap barang dalam inventaris.

Kolom:

- a. Nomor Seri Barang (nomor\_seri\_barang): Tipe data: Varchar(20). Deskripsi: Kunci utama yang secara unik mengidentifikasi setiap barang dalam inventaris.



- b. Tanggal Kadaluwarsa (tanggal\_kadaluwarsa): Tipe data: Date. Deskripsi: Menyimpan tanggal kadaluwarsa barang jika berlaku.
- c. ID Inventaris (id\_inventaris): Tipe data: Integer. Deskripsi: Kunci asing (FK) yang mengacu pada kolom "id\_inventaris" dalam tabel "Inventaris". Menyediakan hubungan antara data detail barang dengan inventaris barang yang terkait.

#### 11. Tabel jenis\_barang

Deskripsi: Tabel ini digunakan untuk menyimpan informasi tentang jenis-jenis barang dalam inventaris.

Kolom:

- a. ID Jenis Barang (id\_jenis\_barang): Tipe data: Integer. Deskripsi: Kunci utama yang secara unik mengidentifikasi setiap jenis barang.
- b. Nomor Seri Barang (nomor\_seri\_barang): Tipe data: Varchar(20). Deskripsi: Kunci asing (FK) yang mengacu pada kolom "nomor\_seri\_barang" dalam tabel "Inventaris". Menyediakan hubungan antara data jenis barang dengan barang yang terkait.
- c. Jenis Barang (jenis\_barang): Tipe data: Varchar(50). Deskripsi: Menyimpan informasi tentang jenis barang dalam inventaris.
- d. Keterangan Jenis Barang (keterangan\_jenis\_barang): Tipe data: Text. Deskripsi: Menyimpan keterangan atau deskripsi tambahan tentang jenis barang.

#### 12. Tabel administrasi\_dan\_pembayaran

Deskripsi: Tabel ini digunakan untuk mencatat informasi administrasi dan pembayaran terkait inventaris dan pasien.

Kolom:

- a. ID Administrasi (id\_administrasi): Tipe data: Integer. Deskripsi: Kunci utama yang secara unik mengidentifikasi setiap administrasi dan pembayaran.
- b. ID Inventaris (id\_inventaris): Tipe data: Integer. Deskripsi: Kunci asing (FK) yang mengacu pada kolom "id\_inventaris" dalam tabel "Inventaris". Menyediakan hubungan antara data administrasi dan pembayaran dengan inventaris barang yang terkait.

- c. ID Pasien (id\_pasien): Tipe data: Integer. Deskripsi: Kunci asing (FK) yang mengacu pada kolom "id\_pasien" dalam tabel "Pasien". Menyediakan hubungan antara data administrasi dan pembayaran dengan pasien yang terkait.
- d. Tanggal Pembayaran (tanggal\_pembayaran): Tipe data: Date. Deskripsi: Menyimpan tanggal pembayaran administrasi.
- e. Total Pembayaran (total\_pembayaran): Tipe data: Decimal(10,2). Deskripsi: Menyimpan informasi tentang total pembayaran administrasi yang harus dilakukan.

### 13. Tabel detail\_pembayaran

Deskripsi: Tabel ini digunakan untuk menyimpan informasi detail pembayaran terkait administrasi dan pembayaran.

Kolom:

- a. Nomor Referensi Pembayaran (nomor\_referensi\_pembayaran): Tipe data: Varchar(20). Deskripsi: Kunci utama yang secara unik mengidentifikasi setiap referensi pembayaran.
- b. Metode Pembayaran (metode\_pembayaran): Tipe data: Varchar(50). Deskripsi: Menyimpan informasi tentang metode pembayaran yang digunakan, misalnya, kartu kredit atau transfer bank.
- c. Status Pembayaran (status\_pembayaran): Tipe data: Varchar(20). Deskripsi: Menyimpan status pembayaran, seperti "Berhasil" atau "Belum Lunas".
- d. ID Administrasi (id\_administrasi): Tipe data: Integer. Deskripsi: Kunci asing (FK) yang mengacu pada kolom "id\_administrasi" dalam tabel "Administrasi\_dan\_Pembayaran". Menyediakan hubungan antara data detail pembayaran dengan administrasi yang terkait.

### 14. Tabel tes\_lab

Deskripsi: Tabel ini digunakan untuk menyimpan informasi tentang tes laboratorium yang dilakukan pada pasien oleh tenaga medis.

Kolom:

- a. ID Tes Lab (id\_tes\_lab): Tipe data: Integer. Deskripsi: Kunci utama yang secara unik mengidentifikasi setiap tes laboratorium.

- b. ID Pasien (id\_pasien): Tipe data: Integer. Deskripsi: Kunci asing (FK) yang mengacu pada kolom "id\_pasien" dalam tabel "Pasien". Menyediakan hubungan antara data tes laboratorium dengan pasien yang bersangkutan.
- c. ID Tenaga Medis (id\_tenaga\_medis): Tipe data: Integer. Deskripsi: Kunci asing (FK) yang mengacu pada kolom "id\_tenaga\_medis" dalam tabel "Tenaga\_Medis". Menyediakan hubungan antara data tes laboratorium dengan tenaga medis yang melakukan tes.
- d. Tanggal Tes Lab (tanggal\_tes\_lab): Tipe data: Date. Deskripsi: Menyimpan tanggal ketika tes laboratorium dilakukan.
- e. Hasil Tes Lab (hasil\_tes\_lab): Tipe data: Text. Deskripsi: Menyimpan hasil atau informasi dari tes laboratorium yang dilakukan.

#### 15. Tabel metode\_tes\_lab

Deskripsi: Tabel ini digunakan untuk menyimpan informasi tentang metode tes laboratorium yang digunakan pada tes laboratorium tertentu.

Kolom:

- a. ID Metode Tes Lab (id\_metode\_tes\_lab): Tipe data: Integer. Deskripsi: Kunci utama yang secara unik mengidentifikasi setiap metode tes laboratorium.
- b. ID Tes Lab (id\_tes\_lab): Tipe data: Integer. Deskripsi: Kunci asing (FK) yang mengacu pada kolom "id\_tes\_lab" dalam tabel "Tes\_Lab". Menyediakan hubungan antara data metode tes laboratorium dengan tes laboratorium yang terkait.
- c. Jenis Tes Lab (jenis\_tes\_lab): Tipe data: Varchar(50). Deskripsi: Menyimpan informasi tentang jenis tes laboratorium yang dilakukan.
- d. Metode Tes Lab (metode\_tes\_lab): Tipe data: Text. Deskripsi: Menyimpan informasi tentang metode atau teknik yang digunakan dalam tes laboratorium tersebut.

#### 16. Tabel appointment

Deskripsi: Tabel ini digunakan untuk menyimpan informasi tentang janji temu (appointment) antara pasien dan tenaga medis.

Kolom:

- a. ID Appointment (id\_appointment): Tipe data: Integer. Deskripsi: Kunci utama yang secara unik mengidentifikasi setiap appointment.

- b. ID Pasien (id\_pasien): Tipe data: Integer. Deskripsi: Kunci asing (FK) yang mengacu pada kolom "id\_pasien" dalam tabel "Pasien". Menyediakan hubungan antara data appointment dengan pasien yang bersangkutan.
- c. Jenis Appointment (jenis\_appointment): Tipe data: Varchar(50). Deskripsi: Menyimpan informasi tentang jenis appointment, seperti pemeriksaan rutin atau konsultasi medis.
- d. Tanggal Mulai Appointment (tanggal\_mulai\_appointment): Tipe data: DateTime. Deskripsi: Menyimpan tanggal dan waktu mulai dari appointment.
- e. Tanggal Selesai Appointment (tanggal\_selesai\_appointment): Tipe data: DateTime. Deskripsi: Menyimpan tanggal dan waktu selesai dari appointment.

#### 17. Tabel detail\_appointment

Deskripsi: Tabel ini digunakan untuk menyimpan informasi detail tentang appointment, termasuk tenaga medis yang terlibat.

Kolom:

- a. ID Detail Appointment (id\_detail\_appointment): Tipe data: Integer. Deskripsi: Kunci utama yang secara unik mengidentifikasi setiap detail appointment.
- b. ID Appointment (id\_appointment): Tipe data: Integer. Deskripsi: Kunci asing (FK) yang mengacu pada kolom "id\_appointment" dalam tabel "Appointment". Menyediakan hubungan antara data detail appointment dengan appointment yang terkait.
- c. ID Tenaga Medis (id\_tenaga\_medis): Tipe data: Integer. Deskripsi: Kunci asing (FK) yang mengacu pada kolom "id\_tenaga\_medis" dalam tabel "Tenaga\_Medis". Menyediakan informasi tentang tenaga medis yang terlibat dalam appointment.

#### 18. Tabel lokasi\_appointment

Deskripsi: Tabel ini digunakan untuk menyimpan informasi tentang lokasi dan waktu appointment.

Kolom:

- a. Nomor Ruangan (nomor\_ruangan): Tipe data: Varchar(10). Deskripsi: Kunci utama yang secara unik mengidentifikasi setiap nomor ruangan.
- b. ID Detail Appointment (id\_detail\_appointment): Tipe data: Integer. Deskripsi: Kunci asing (FK) yang mengacu pada kolom "id\_detail\_appointment" dalam

tabel "Detail\_Appointment". Menyediakan hubungan antara data lokasi appointment dengan detail appointment yang terkait.

- c. Jam Appointment (jam\_appointment): Tipe data: Time. Deskripsi: Menyimpan informasi tentang jam atau waktu appointment.

## **BAB IV**

### **DESKRIPSI FUNGSIONALITAS**

#### **4.1 Fungsionalitas Sistem**

1. Manajemen Pasien:
  - a. Pendaftaran pasien baru dengan informasi pribadi, riwayat penyakit, alergi, dan informasi asuransi.
  - b. Pembaruan informasi pasien secara berkala.
  - c. Pengelolaan jadwal rawat inap, rawat jalan, dan prosedur medis.
2. Manajemen Tenaga Medis:
  - a. Pemantauan Kinerja dan Penggajian
  - b. Pengelompokan dan Klasifikasi Tenaga Medis
3. Rekam Medis Elektronik:
  - a. Pengarsipan data medis elektronik pasien, termasuk catatan diagnosa, hasil tes, dan riwayat pengobatan.
  - b. Akses terbatas sesuai dengan peran (dokter, perawat).
4. Manajemen Inventaris:
  - a. Pemantauan stok obat, alat medis, peralatan rumah sakit.
  - b. Pengelolaan stok secara otomatis.
5. Manajemen Appointment:
  - a. Penjadwalan, pembaruan, dan manajemen shift dokter dan perawat.
  - b. Informasi kontak dan spesialisasi masing-masing tenaga medis.
6. Administrasi dan Pembayaran:
  - a. Pembuatan tagihan dan manajemen data asuransi pasien.
  - b. Penerimaan dan pencatatan pembayaran.
7. Manajemen Laboratorium:
  - a. Pencatatan dan pelacakan hasil tes laboratorium.
  - b. Integrasi dengan rekam medis untuk informasi yang lebih komprehensif.

## BAB V

### IMPLEMENTASI SQL

#### 5.1 Query Create Database and Table

```
DROP DATABASE IF EXISTS medcare;
CREATE DATABASE medcare;
USE medcare;

DROP TABLE IF EXISTS pasien;
CREATE TABLE pasien(
    id_pasien INT PRIMARY KEY,
    nama_pasien VARCHAR(80),
    tanggal_lahir TIMESTAMP,
    jenis_kelamin VARCHAR(80)
);

DROP TABLE IF EXISTS asuransi;
CREATE TABLE asuransi(
    nomor_asuransi VARCHAR(80) PRIMARY KEY,
    nama_perusahaan_asuransi VARCHAR(80),
    id_pasien INT,
    FOREIGN KEY (id_pasien) REFERENCES pasien(id_pasien)
);

DROP TABLE IF EXISTS asuransi_pasien;
CREATE TABLE asuransi_pasien(
    id_pasien INT,
    nomor_asuransi VARCHAR(80),
    FOREIGN KEY (nomor_asuransi) REFERENCES asuransi(nomor_asuransi)
);

DROP TABLE IF EXISTS tenaga_medis;
CREATE TABLE tenaga_medis(
    id_tenaga_medis INT PRIMARY KEY,
    nama_tenaga_medis VARCHAR(80)
);

DROP TABLE IF EXISTS spesialisasi_tenaga_medis;
CREATE TABLE spesialisasi_tenaga_medis(
    id_spesialisasi INT PRIMARY KEY,
    id_tenaga_medis INT,
    jenis_pekerjaan VARCHAR(80),
    spesialisasi_tenaga_medis VARCHAR(80),
    FOREIGN KEY (id_tenaga_medis) REFERENCES tenaga_medis(id_tenaga_medis)
```

```

);

DROP TABLE IF EXISTS kontak_dan_gaji;
CREATE TABLE kontak_dan_gaji(
    kontak_tenaga_medis VARCHAR(80) PRIMARY KEY,
    id_tenaga_medis INT,
    gaji_pokok DECIMAL(10,2),
    gaji_bonus DECIMAL(10,2),
    FOREIGN KEY (id_tenaga_medis) REFERENCES tenaga_medis(id_tenaga_medis)
);

DROP TABLE IF EXISTS catatan_pasien;
CREATE TABLE catatan_pasien(
    id_rekam_medis INT PRIMARY KEY,
    id_pasien INT,
    id_tenaga_medis INT,
    tanggal_catatan_medis TIMESTAMP,
    FOREIGN KEY (id_pasien) REFERENCES pasien(id_pasien),
    FOREIGN KEY (id_tenaga_medis) REFERENCES tenaga_medis(id_tenaga_medis)
);

DROP TABLE IF EXISTS diagnosa;
CREATE TABLE diagnosa(
    id_diagnosa INT PRIMARY KEY,
    id_rekam_medis INT,
    diagnosa VARCHAR(80),
    deskripsi_diagnosa TEXT,
    metode_pengobatan VARCHAR(80),
    FOREIGN KEY (id_rekam_medis) REFERENCES catatan_pasien(id_rekam_medis)
);

DROP TABLE IF EXISTS inventaris;
CREATE TABLE inventaris(
    id_inventaris INT PRIMARY KEY,
    nama_barang VARCHAR(80),
    jumlah_stok_barang DECIMAL(10,8),
    lokasi_penyimpanan_barang VARCHAR(80)
);

DROP TABLE IF EXISTS detail_barang;
CREATE TABLE detail_barang(
    nomor_seri_barang VARCHAR(80) PRIMARY KEY,
    tanggal_kadaluwarsa DATETIME,
    id_inventaris INT,
    FOREIGN KEY (id_inventaris) REFERENCES inventaris(id_inventaris)
);

```



```

);

DROP TABLE IF EXISTS jenis_barang;
CREATE TABLE jenis_barang(
    id_jenis_barang INT PRIMARY KEY,
    nomor_seri_barang VARCHAR(80),
    jenis_barang VARCHAR(80),
    keterangan_jenis_barang TEXT,
    FOREIGN KEY (nomor_seri_barang) REFERENCES
detail_barang(nomor_seri_barang)
);

DROP TABLE IF EXISTS administrasi_dan_pembayaran;
CREATE TABLE administrasi_dan_pembayaran(
    id_administrasi INT PRIMARY KEY,
    id_inventaris INT,
    id_pasien INT,
    tanggal_pembayaran TIMESTAMP,
    total_pembayaran DECIMAL(10,2),
    FOREIGN KEY (id_pasien) REFERENCES pasien(id_pasien),
    FOREIGN KEY (id_inventaris) REFERENCES inventaris(id_inventaris)
);

DROP TABLE IF EXISTS detail_pembayaran;
CREATE TABLE detail_pembayaran(
    nomor_referensi_pembayaran VARCHAR(80) PRIMARY KEY,
    metode_pembayaran VARCHAR(80),
    status_pembayaran VARCHAR(80),
    id_administrasi INT,
    FOREIGN KEY (id_administrasi) REFERENCES
administrasi_dan_pembayaran(id_administrasi)
);

DROP TABLE IF EXISTS tes_lab;
CREATE TABLE tes_lab(
    id_tes_lab INT PRIMARY KEY,
    id_pasien INT,
    id_tenaga_medis INT,
    tanggal_tes_lab DATETIME,
    hasil_tes_lab VARCHAR(80),
    FOREIGN KEY (id_pasien) REFERENCES pasien(id_pasien),
    FOREIGN KEY (id_tenaga_medis) REFERENCES tenaga_medis(id_tenaga_medis)
);

DROP TABLE IF EXISTS metode_tes_lab;

```

```
CREATE TABLE metode_tes_lab(  
    id_metode_tes_lab INT PRIMARY KEY,  
    id_tes_lab INT,  
    jenis_tes_lab VARCHAR(80),  
    metode_tes_lab VARCHAR(80),  
    FOREIGN KEY (id_tes_lab) REFERENCES tes_lab(id_tes_lab)  
);  
  
DROP TABLE IF EXISTS appointment;  
CREATE TABLE appointment(  
    id_appointment INT PRIMARY KEY,  
    id_pasien INT,  
    jenis_appointment VARCHAR(80),  
    tanggal_mulai_appointment DATETIME,  
    tanggal_selesai_appointment DATETIME,  
    FOREIGN KEY (id_pasien) REFERENCES pasien(id_pasien)  
);  
  
DROP TABLE IF EXISTS detail_appointment;  
CREATE TABLE detail_appointment(  
    id_detail_appointment INT PRIMARY KEY,  
    id_appointment INT,  
    id_tenaga_medis INT,  
    FOREIGN KEY (id_appointment) REFERENCES appointment(id_appointment),  
    FOREIGN KEY (id_tenaga_medis) REFERENCES tenaga_medis(id_tenaga_medis)  
);  
  
DROP TABLE IF EXISTS lokasi_appointment;  
CREATE TABLE lokasi_appointment(  
    nomor_ruangan INT PRIMARY KEY,  
    id_detail_appointment INT,  
    jam_appointment TIMESTAMP,  
    FOREIGN KEY (id_detail_appointment) REFERENCES  
detail_appointment(id_detail_appointment)  
);
```

## 5.2 Query Insert Dump Data

```
insert into Pasien (id_pasien, nama_pasien, tanggal_lahir, jenis_kelamin)
values (1, 'Emanuele Linnit', '2009-12-27', 'Male');
insert into asuransi (nomor_asuransi, nama_perusahaan_asuransi, id_pasien)
values ('8RT9RE8WN30', 'Stroman-Hills', 1);
insert into Asuransi_Pasien (id_pasien, nomor_asuransi) values (1,
'8RT9RE8WN30');
insert into Tenaga_Medis (id_tenaga_medis, nama_tenaga_medis) values (1,
'Hogan Bampfield');
insert into spesialisasi_tenaga_medis (id_spesialisasi, id_tenaga_medis,
jenis_pekerjaan, spesialisasi_tenaga_medis) values (1, 766, 'Nurse',
'Urology');
insert into Kontak_dan_Gaji (kontak_tenaga_medis, id_tenaga_medis,
gaji_pokok, gaji_bonus) values ('(317) 9161888', 1, 2008.04, 5888.1);
insert into catatan_pasien (id_rekam_medis, id_pasien, id_tenaga_medis,
tanggal_catatan_medis) values (1, 965, 356, '2009-07-16');
insert into diagnosa (id_diagnosa, id_rekam_medis, diagnosa,
deskripsi_diagnosa, metode_pengobatan) values (1, 314, 'strep throat',
'Rash', 'acupuncture');
insert into inventaris (id_inventaris, nama_barang, jumlah_stok_barang,
lokasi_penyimpanan_barang) values (1, 'stethoscope', 64, 'Gudang A');
insert into detail_barang (nomor_seri_barang, tanggal_kadaluwarsa,
id_inventaris) values ('0812819691', '2022-02-15', 337);
insert into jenis_barang (id_jenis_barang, nomor_seri_barang, jenis_barang,
keterangan_jenis_barang) values (1, '4449866460', 'disinfectant', 'Cotton
swabs');
insert into administrasi_dan_pembayaran (id_administrasi, id_inventaris,
id_pasien, tanggal_pembayaran, total_pembayaran) values (1, 378, 69,
'2023-02-17', 3236.48);
insert into detail_pembayaran (nomor_referensi_pembayaran,
metode_pembayaran, status_pembayaran, id_administrasi) values
('80-406-6105', 'Cash', 'pending', 750);
insert into tes_lab (id_tes_lab, id_pasien, id_tenaga_medis,
tanggal_tes_lab, hasil_tes_lab) values (1, 852, 835, '2017-10-30',
'negative');
```

Query lengkapnya dapat dilihat pada link github berikut:

<https://github.com/J0see1/FP-SBD.git>

## BAB VI

### QUERY BERDASARKAN FUNGSIONALITAS

#### 6.1 Transaction dan Rollback

```
START TRANSACTION;

SELECT tenaga_medis.nama_tenaga_medis,
spesialisasi_tenaga_medis.jenis_pekerjaan,
kontak_dan_gaji.kontak_tenaga_medis
FROM tenaga_medis
INNER JOIN spesialisasi_tenaga_medis ON tenaga_medis.id_tenaga_medis =
spesialisasi_tenaga_medis.id_tenaga_medis
INNER JOIN kontak_dan_gaji ON tenaga_medis.id_tenaga_medis =
kontak_dan_gaji.id_tenaga_medis
GROUP BY tenaga_medis.nama_tenaga_medis,
spesialisasi_tenaga_medis.jenis_pekerjaan;

DELETE FROM spesialisasi_tenaga_medis WHERE jenis_pekerjaan="Doctor";

SELECT
    subquery.nama_tenaga_medis,
    subquery.jenis_pekerjaan,
    subquery.total_gaji
FROM (
    SELECT
        tenaga_medis.nama_tenaga_medis,
        spesialisasi_tenaga_medis.jenis_pekerjaan,
        SUM(kontak_dan_gaji.gaji_pokok + kontak_dan_gaji.gaji_bonus) AS
total_gaji
    FROM
        tenaga_medis
    INNER JOIN
        spesialisasi_tenaga_medis ON tenaga_medis.id_tenaga_medis =
spesialisasi_tenaga_medis.id_tenaga_medis
    INNER JOIN
        kontak_dan_gaji ON tenaga_medis.id_tenaga_medis =
kontak_dan_gaji.id_tenaga_medis
    GROUP BY
        tenaga_medis.nama_tenaga_medis,
spesialisasi_tenaga_medis.jenis_pekerjaan
    HAVING
        total_gaji < 5000
```

```

        ORDER BY total_gaji ASC
    ) AS subquery;

ROLLBACK;

SELECT
    tenaga_medis.nama_tenaga_medis,
    spesialisasi_tenaga_medis.jenis_pekerjaan,
    spesialisasi_tenaga_medis.spesialisasi_tenaga_medis
FROM
    tenaga_medis
INNER JOIN
    spesialisasi_tenaga_medis ON tenaga_medis.id_tenaga_medis =
    spesialisasi_tenaga_medis.id_tenaga_medis
INNER JOIN
    kontak_dan_gaji ON tenaga_medis.id_tenaga_medis =
    kontak_dan_gaji.id_tenaga_medis
WHERE spesialisasi_tenaga_medis.jenis_pekerjaan = 'Doctor' AND
    spesialisasi_tenaga_medis.spesialisasi_tenaga_medis = 'Orthopedics'
GROUP BY
    tenaga_medis.nama_tenaga_medis

```

## 6.2 View

```

-- Membuat view dengan menggunakan beberapa elemen
CREATE VIEW view_tenaga_medis_summary AS
SELECT
    tm.id_tenaga_medis,
    tm.nama_tenaga_medis,
    COUNT(stm.id_spesialisasi) AS jumlah_spesialisasi,
    SUM(kg.gaji_pokok + kg.gaji_bonus) AS total_gaji
FROM
    tenaga_medis tm
LEFT JOIN spesialisasi_tenaga_medis stm ON tm.id_tenaga_medis =
    stm.id_tenaga_medis
LEFT JOIN kontak_dan_gaji kg ON tm.id_tenaga_medis = kg.id_tenaga_medis
WHERE
    kg.gaji_pokok IS NOT NULL AND kg.gaji_bonus IS NOT NULL
GROUP BY
    tm.id_tenaga_medis, tm.nama_tenaga_medis
HAVING
    COUNT(stm.id_spesialisasi) > 1

```

```

ORDER BY
    total_gaji DESC;

-- cara jalaninnya:

SELECT * FROM view_tenaga_medis_summary;

SELECT *
FROM view_tenaga_medis_summary
WHERE jumlah_spesialisasi > 3;

SELECT AVG(total_gaji) AS rata_rata_gaji
FROM view_tenaga_medis_summary;

```

### 6.3 Stored Procedure

```

DELIMITER //

CREATE PROCEDURE GetPatients(
    IN diagnosis_name VARCHAR(80),
    IN lab_result VARCHAR(80),
    IN min_lab_count INT
)
BEGIN
    SELECT
        p.id_pasien,
        p.nama_pasien,
        COUNT(tl.id_tes_lab) AS total_lab_count
    FROM pasien p
    JOIN catatan_pasien cp ON p.id_pasien = cp.id_pasien
    JOIN tes_lab tl ON cp.id_rekam_medis = tl.id_pasien
    JOIN diagnosa d ON cp.id_rekam_medis = d.id_rekam_medis
    WHERE d.diagnosa = diagnosis_name AND tl.hasil_tes_lab = lab_result
    GROUP BY p.id_pasien, p.nama_pasien
    HAVING total_lab_count >= min_lab_count
    ORDER BY total_lab_count DESC;
END //

DELIMITER ;

CALL GetPatients('gastritis', 'positive', 1);

```

## 6.4 Stored Function

```
DELIMITER //
```

```
CREATE FUNCTION calculateAverageAge(gender_filter VARCHAR(6))
RETURNS DECIMAL(10,2)
BEGIN
    DECLARE total_umur DECIMAL(10,2);
    DECLARE jumlah_pasien INT;

    SELECT SUM(TIMESTAMPDIFF(YEAR, tanggal_lahir, NOW())) INTO
total_umur
    FROM pasien
    WHERE jenis_kelamin = gender_filter AND YEAR(tanggal_lahir) >=
1970;

    SELECT COUNT(*) INTO jumlah_pasien
    FROM pasien
    WHERE jenis_kelamin = gender_filter AND YEAR(tanggal_lahir) >=
1970;

    IF jumlah_pasien > 0 THEN
        RETURN total_umur / jumlah_pasien;
    ELSE
        RETURN NULL;
    END IF;
END //
```

```
DELIMITER ;
```

```
SELECT calculateAverageAge('Male') AS average_age_male;
```

## 6.5 Trigger

```
-- Membuat trigger setelah pembaruan pada detail_appointment
DELIMITER //
```

```
CREATE TRIGGER after_update_detail_appointment
AFTER UPDATE ON detail_appointment
FOR EACH ROW
BEGIN
    -- Variabel untuk menyimpan total jumlah appointment untuk pasien
tertentu
    DECLARE total_appointments INT;
```

```

-- Menggunakan subquery untuk mendapatkan total jumlah appointment
untuk pasien yang sama
SELECT COUNT(*) INTO total_appointments
FROM appointment a
JOIN detail_appointment da ON a.id_appointment = da.id_appointment
WHERE a.id_pasien = (SELECT id_pasien FROM appointment WHERE
id_appointment = NEW.id_appointment);

-- Menambahkan log ke dalam tabel audit_log berdasarkan kondisi
IF NEW.id_tenaga_medis IS NOT NULL AND total_appointments > 1 THEN
    INSERT INTO audit_log (log_message, total_appointments)
    VALUES ('Detail appointment diperbarui dengan tenaga medis dan
pasien yang sama memiliki lebih dari 1 appointment',
total_appointments);
END IF;
END;
//
DELIMITER ;

-- membuat tabel audit_log
DROP TABLE IF EXISTS audit_log;
CREATE TABLE audit_log (
    id_log INT AUTO_INCREMENT PRIMARY KEY,
    log_message VARCHAR(255),
    log_timestamp TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    total_stok INT;
);

-- cara jalaninnya:
-- Contoh pernyataan pembaruan untuk menjalankan trigger
UPDATE detail_appointment SET id_tenaga_medis = 001 WHERE
id_detail_appointment = 1;
-- Pernyataan SELECT untuk melihat log di tabel audit_log
SELECT * FROM audit_log;

```



