Nama: Zhafira Hasifaturrahmah

NIM: 40011423650271

Sistem Informasi Akuntansi (SIA) adalah sistem yang yang mengumpulkan, mencatat, menyimpan, dan memproses data tentang peristiwa, sumber daya, dan agen untuk menghasikan informasi bagi pengambil keputusan, serta memberikan pengendalian yang memadai untuk memastikan bahwa sumber daya entitas tersedia bila diperlukan dan akurat dan dapat diandalkan. Sistem (terdiri dari subsistem) adalah seperangkat komponen yang saling terikat dan berinteraksi untuk memaksimalkan tercapainya tujuan organisasi.

**System Documentation Techniques** merujuk pada berbagai metode atau pendekatan yang digunakan untuk mendokumentasikan sistem dalam suatu organisasi. Dokumentasi ini bertujuan untuk memberikan pemahaman yang jelas mengenai sistem, proses, dan prosedur yang digunakan, serta membantu pengembangan, pemeliharaan, dan pengelolaan sistem. Teknik ini penting dalam berbagai bidang seperti pengembangan perangkat lunak, manajemen proyek, dan audit sistem informasi. System documentation Techniques merujuk pada dokumentasi atau pengumpulan data dan informasi menggunakan flowchart agar lebih akurat, yang menggambarkan alur system.

**Jenis-jenis System Documentation Techniques**

1. **Narrative Documentation (Dokumentasi Naratif)**:
   * Berupa deskripsi tertulis atau penjelasan rinci tentang bagaimana sistem bekerja.
   * Menjelaskan proses-proses bisnis secara kronologis.
   * Contoh: Manual pengguna, pedoman kebijakan, atau laporan tertulis.
2. **Flowcharts (Diagram Alir)**:
   * Menggambarkan langkah-langkah atau proses dalam sistem dengan simbol dan panah.
   * Membantu memvisualisasikan alur kerja dan proses logis.
   * Jenis-jenis flowchart meliputi:
     + **System Flowchart** – Menggambarkan alur sistem secara keseluruhan.
     + **Program Flowchart** – Menjelaskan alur logis dari program komputer.
     + **Document Flowchart** – Menggambarkan alur dokumen dalam organisasi.
3. **Data Flow Diagrams (DFD)**:
   * Memvisualisasikan aliran data antara komponen dalam sistem.
   * Terdiri dari entitas eksternal, proses, penyimpanan data, dan aliran data.
   * DFD sering digunakan dalam analisis dan perancangan sistem.
4. **Entity-Relationship Diagrams (ERD)**:
   * Menggambarkan hubungan antara entitas dalam basis data.
   * Membantu dalam perancangan database relasional.
   * Menggunakan simbol untuk menunjukkan entitas, atribut, dan relasi antar entitas.
5. **Unified Modeling Language (UML)**:
   * Koleksi diagram untuk memodelkan sistem perangkat lunak.
   * Contoh diagram UML meliputi:
     + **Class Diagram** – Menggambarkan struktur kelas dan hubungan antar kelas.
     + **Use Case Diagram** – Menunjukkan interaksi antara pengguna dan sistem.
     + **Sequence Diagram** – Menunjukkan urutan interaksi antar objek.
6. **Decision Tables and Decision Trees**:
   * **Decision Tables** – Tabel yang menunjukkan kombinasi kondisi dan tindakan yang diambil berdasarkan kondisi tersebut.
   * **Decision Trees** – Diagram bercabang yang menggambarkan proses pengambilan keputusan.
7. **Pseudocode**:
   * Deskripsi logika program menggunakan bahasa sederhana (mirip dengan bahasa pemrograman).
   * Membantu dalam menulis kode sebelum implementasi sebenarnya.
8. **System Architecture Diagrams**:
   * Menggambarkan komponen dan hubungan antar komponen dalam arsitektur sistem.
   * Contoh: Diagram jaringan, diagram infrastruktur TI.

Berikut adalah struktur database penjualan yang lebih sederhana dengan 3 tabel: produk, pelanggan, dan penjualan:

**1. Tabel produk (untuk menyimpan informasi tentang produk)**

CREATE TABLE produk (

id\_produk INT PRIMARY KEY AUTO\_INCREMENT,

nama\_produk VARCHAR(100) NOT NULL,

harga DECIMAL(10, 2) NOT NULL,

stok INT NOT NULL

);

**2. Tabel pelanggan (untuk menyimpan informasi tentang pelanggan)**

CREATE TABLE pelanggan (

id\_pelanggan INT PRIMARY KEY AUTO\_INCREMENT,

nama\_pelanggan VARCHAR(100) NOT NULL,

telepon VARCHAR(20),

alamat VARCHAR(255)

);

**3. Tabel penjualan (untuk menyimpan informasi transaksi penjualan, termasuk produk yang dijual dan pelanggan yang melakukan pembelian)**

CREATE TABLE penjualan (

id\_penjualan INT PRIMARY KEY AUTO\_INCREMENT,

id\_pelanggan INT,

id\_produk INT,

tanggal\_penjualan DATE NOT NULL,

jumlah INT NOT NULL,

total\_harga DECIMAL(10, 2),

FOREIGN KEY (id\_pelanggan) REFERENCES pelanggan(id\_pelanggan),

FOREIGN KEY (id\_produk) REFERENCES produk(id\_produk)

);

**Contoh Query untuk Menambahkan Data**

Berikut adalah contoh query untuk menambahkan data ke tabel-tabel tersebut:

**Menambahkan Produk**

INSERT INTO produk (nama\_produk, harga, stok)

VALUES ('Produk A', 100000, 20);

**Menambahkan Pelanggan**

INSERT INTO pelanggan (nama\_pelanggan, telepon, alamat)

VALUES ('John Doe', '081234567890', 'Jl. Mawar No. 1');

**Menambahkan Penjualan**

INSERT INTO penjualan (id\_pelanggan, id\_produk, tanggal\_penjualan, jumlah, total\_harga)

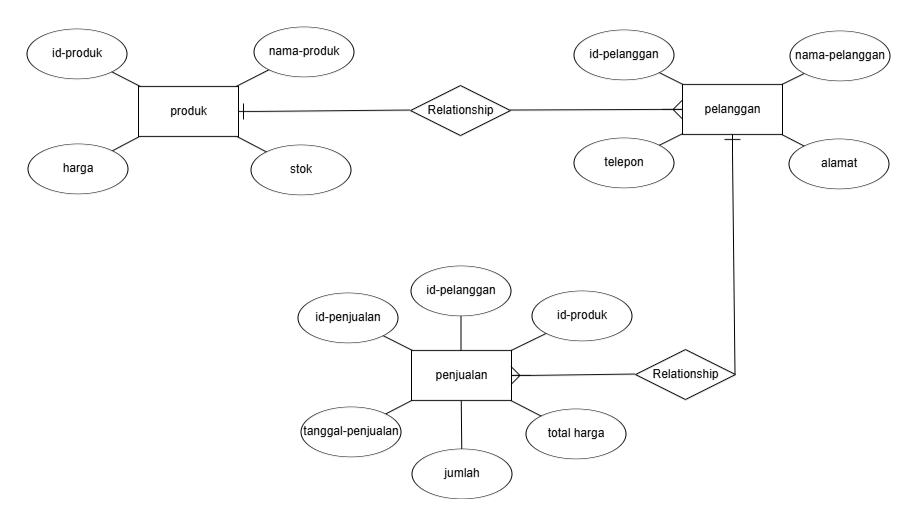
VALUES (1, 1, '2024-10-25', 2, 200000);

Dalam model 3 tabel ini:

* Tabel produk menyimpan informasi produk.
* Tabel pelanggan menyimpan informasi pelanggan.
* Tabel penjualan menyimpan transaksi penjualan yang menghubungkan pelanggan, produk, dan jumlah produk yang dibeli dalam satu transaksi.

Ini adalah struktur yang cukup sederhana dan cocok untuk database penjualan dasar.

**ERD Yang Dihasilkan Dari Ketiga Tabel Di Atas**



Untuk hubungan kardinalitas pada ERD dari ketiga tabel di atas, berikut adalah penjelasan hubungan antara tabel Produk, Pelanggan, dan Penjualan:

1. **Hubungan antara Pelanggan dan Penjualan:**
   * Kardinalitas: **One-to-Many** (1:N)
   * Artinya, **satu pelanggan** dapat melakukan **banyak penjualan**, tetapi **satu penjualan** hanya bisa dilakukan oleh **satu pelanggan**.
   * Pada ERD, ini digambarkan dengan satu sisi Pelanggan menghubungkan banyak entitas di Penjualan.
2. **Hubungan antara Produk dan Penjualan:**
   * Kardinalitas: **One-to-Many** (1:N)
   * Artinya, **satu produk** dapat dijual dalam **banyak transaksi penjualan**, tetapi **satu penjualan** hanya bisa memiliki **satu produk**.
   * Pada ERD, ini digambarkan dengan satu sisi Produk menghubungkan banyak entitas di Penjualan.

**Visualisasi Kardinalitas**

* **Pelanggan** (1) → (N) **Penjualan**
* **Produk** (1) → (N) **Penjualan**

Ini berarti bahwa setiap pelanggan dan produk dapat muncul di banyak entitas penjualan, tetapi setiap entitas penjualan terkait hanya dengan satu pelanggan dan satu produk.