Berikut adalah **5 analisis relasi** dari database `skaven_rpl` beserta alasan pemilihan **kardinalitas**, **primary key**, dan **foreign key**.

---

1. Relasi `siswa` → `transaksi` (One-to-Many)

- **Kardinalitas**: **One-to-Many**
  - **Alasan**: Satu siswa **dapat melakukan banyak transaksi**, tetapi satu transaksi hanya milik satu siswa tertentu.
- **Primary Key & Foreign Key**:
  - **Primary Key**: `id_siswa` di tabel `siswa` karena setiap siswa memiliki ID masing masing.
  - **Foreign Key**: `id_siswa` di tabel `transaksi`, karena setiap transaksi harus terhubung dengan satu siswa tertentu.
- **SQL**:

```
CREATE TABLE transaksi (
    id_transaksi INT PRIMARY KEY AUTO_INCREMENT,
    id_siswa INT NOT NULL,
    tanggal_transaksi DATE NOT NULL,
    total_bayar DECIMAL(10,2) NOT NULL,
    FOREIGN KEY (id_siswa) REFERENCES siswa(id_siswa) ON DELETE CASCADE
);
```

---

2. Relasi `siswa` ↔ `eskul` melalui `eskul_siswa` (Many-to-Many)

- **Kardinalitas**: **Many-to-Many**
  - **Alasan**: Satu siswa bisa mengikuti **banyak ekstrakurikuler**, dan satu ekstrakurikuler juga bisa diikuti oleh **banyak siswa**.
- **Primary Key & Foreign Key**:
  - **Primary Key**:
    - `id_siswa` di tabel `siswa`, karena setiap siswa memiliki ID unik.
    - `eskul_id` di tabel `eskul`, karena setiap ekstrakurikuler memiliki ID unik.

- **Foreign Key**:
  - `id_siswa` di tabel `eskul_siswa`, sebagai referensi ke tabel `siswa`.
  - `eskul_id` di tabel `eskul_siswa`, sebagai referensi ke tabel `eskul`.
- **SQL**:

```
CREATE TABLE eskul_siswa (
    id_siswa INT NOT NULL,
    eskul_id INT NOT NULL,
    tanggal_gabung DATE NOT NULL,
    PRIMARY KEY (id_siswa, eskul_id),
    FOREIGN KEY (id_siswa) REFERENCES siswa(id_siswa) ON DELETE CASCADE,
    FOREIGN KEY (eskul_id) REFERENCES eskul(eskul_id) ON DELETE CASCADE
);
```

---

3. Relasi `guru` → `mata_pelajaran` (One-to-Many)

- **Kardinalitas**: **One-to-Many**
  - **Alasan**: Satu guru **bisa mengajar banyak mata pelajaran**, tetapi satu mata pelajaran hanya diajarkan oleh **satu guru** tertentu.
- **Primary Key & Foreign Key**:
  - **Primary Key**: `nip` di tabel `guru`, karena setiap guru memiliki nomor induk unik.
  - **Foreign Key**: `nip` di tabel `mata_pelajaran`, karena setiap mata pelajaran harus memiliki seorang guru pengampu.
- **SQL**:

```
CREATE TABLE mata_pelajaran (
    id_mapel INT PRIMARY KEY AUTO_INCREMENT,
    nip INT NOT NULL,
    nama_mapel VARCHAR(100) NOT NULL,
    deskripsi TEXT NOT NULL,
    FOREIGN KEY (nip) REFERENCES guru(nip) ON DELETE CASCADE
);
```

---

4. Relasi `transaksi` → `detail_transaksi` (One-to-Many)

- **Kardinalitas**: **One-to-Many**
  - **Alasan**: Satu transaksi bisa memiliki **banyak item yang dibeli** (detail transaksi), tetapi **satu detail transaksi hanya terkait dengan satu transaksi saja**.
- **Primary Key & Foreign Key**:
  - **Primary Key**: `id_transaksi` di tabel `transaksi`, karena setiap transaksi memiliki ID unik.
  - **Foreign Key**:
    - `id_transaksi` di tabel `detail_transaksi`, karena setiap detail transaksi terkait dengan satu transaksi tertentu.
    - `id_produk` di tabel `detail_transaksi`, karena setiap detail transaksi mencatat produk tertentu yang dibeli.
- **SQL**:

```
CREATE TABLE detail_transaksi (
    id_detail INT PRIMARY KEY AUTO_INCREMENT,
    id_transaksi INT NOT NULL,
    id_produk INT NOT NULL,
    jumlah INT NOT NULL,
    total_harga DECIMAL(10,2) NOT NULL,
    FOREIGN KEY (id_transaksi) REFERENCES transaksi(id_transaksi) ON DELETE
CASCADE,
    FOREIGN KEY (id_produk) REFERENCES produk(id_produk) ON DELETE CASCADE
);
```

---

5. Relasi `event_sekolah` → `pendaftaran` (One-to-Many)

- **Kardinalitas**: **One-to-Many**
  - **Alasan**: Satu event sekolah bisa memiliki **banyak siswa yang mendaftar**, tetapi satu siswa hanya mendaftar untuk **satu event tertentu** dalam satu entri.
- **Primary Key & Foreign Key**:
  - **Primary Key**: `id_event` di tabel `event_sekolah`, karena setiap event memiliki ID unik.
  - **Foreign Key**:

    - `id_event` di tabel `pendaftaran`, karena setiap pendaftaran terkait dengan satu event tertentu.
    - `id_siswa` di tabel `pendaftaran`, karena setiap pendaftaran harus mencatat siswa yang mendaftar.
- **SQL**:

```
CREATE TABLE pendaftaran (
    id_daftar INT PRIMARY KEY AUTO_INCREMENT,
    id_event INT NOT NULL,
    id_siswa INT NOT NULL,
    nama_siswa VARCHAR(100) NOT NULL,
    FOREIGN KEY (id_event) REFERENCES event_sekolah(id_event) ON DELETE CASCADE,
    FOREIGN KEY (id_siswa) REFERENCES siswa(id_siswa) ON DELETE CASCADE
);
```

Kesimpulan

1. **Relasi One-to-Many** banyak digunakan untuk data yang memiliki hierarki atau struktur kepemilikan seperti **siswa** → **transaksi**, **guru** → **mata pelajaran**, dan **event** → **pendaftaran**.
2. **Relasi Many-to-Many** memerlukan **tabel perantara**, seperti **siswa** ↔ **eskul** melalui `eskul_siswa`, agar tidak terjadi redundansi.
3. **Pemilihan Foreign Key** tergantung pada tabel yang membutuhkan referensi ke tabel lain, memastikan integritas data tetap terjaga.
4. **Penggunaan** `ON DELETE CASCADE` sangat berguna untuk memastikan data yang berkaitan dihapus secara otomatis jika data induknya dihapus.
   Dengan struktur ini, database tetap **terstruktur, optimal, dan mudah di-maintenance**!

Laporan Aktivitas 13 Februari 2025

No	Nama	Skor Keaktifan	Peran
1	FAROEK AL QAYYUM	3	menganalisis Relasi `siswa` → `transaksi` (One-to-Many)**
2	INDRAWAN	3	menganalisis Relasi `siswa` ↔ `eskul` melalui `eskul_siswa` (Many-to-Many)

No	Nama	Skor Keaktifan	Peran
3	RIVALDO GABRIEL	3	menganalisis Relasi `guru` → `mata_pelajaran` (One-to-Many)
4	VALENTINO	3	menganalisis Relasi `transaksi` → `detail_transaksi` (One-to-Many)
5	IQBAL	3	menganalisis Relasi `event_sekolah` → `pendaftaran` (One-to-Many)
6	MUHAMMAD ARIEL		