DATABASE FUNDAMENTAL

KELAS PROGRAMMING FULL STACK DEVELOPER

MITRA PELATIHAN

Alkademi

2024

Jabar Digital Academy

digitalacademy.jabarprov.go.id

BAB I
DATABASE FUNDAMENTAL.

A. Tujuan

a. Peserta didik dapat memahami apa itu database.
b. Peserta didik dapat memahami sejarah database.
c. Peserta didik dapat memahami schema dalam database.
d. Peserta didik dapat memahami ERD dalam merancang database.
e. Peserta didik dapat membuat relasi antar Entitas.

B. Perlengkapan

a. Modul Bab 1. Database Fundamental.
b. Komputer dengan sistem operasi Windows bila diperlukan

C. Materi

Pernahkah kalian membayangkan bagaimana sebuah aplikasi dapat menyimpan jutaan data dan dapat menampilkan data tersebut dalam waktu yang singkat? Bagaimana *status*, *story*, atau *upload*-an sosial media teman kalian dapat kalian lihat? Atau bahkan sampai konten sosial media yang sudah bertahun-tahun yang lalu tetap ada? Lalu, bagaimana semua data dalam website-website yang kita akses dapat saling berhubungan dan saling melengkapi satu sama lain?

Untuk memenuhi semua kebutuhan penyimpanan, pengolahan, dan penyajian data kita perlu tempat untuk menyimpan semua data tersebut, diciptakanlah teknologi yang sekarang kita kenal dengan nama database.

a. Apa itu Database?

Database adalah sebuah tempat untuk menyimpan sekumpulan data, pada umumnya database dapat menyimpan data baik text, numerik, waktu, dan sebagainya. Data-data ini diatur dengan rapi dalam sistem yang memungkinkan kita untuk mengelolanya sesuai kebutuhan. Kita dapat melakukan beberapa macam operasi di dalam Database

seperti pencarian, penyisipan, pembaruan, dan penghapusan data, serta pengelolaan struktur database.

b. **Sejarah Database.**

Database management system (DBMS) pertama kali dikembangkan pada tahun 1960 oleh *Charles Bachman*, dengan tujuan untuk menyediakan cara menyimpan dan memanipulasi data dalam jumlah besar yang lebih efisien dan terorganisir.

Source : https://repository.aip.org/islandora/object/nbla:294541.

Sistem awal ini didasarkan pada model relasional yang mengatur data kedalam tabel dan mendefinisikan hubungan antar mereka. CODASYL *(Conference of Data System Language)* membentuk dan menstandarisasi model ini. DBMS pertama dan dikomersialkan adalah *Information Management System IBM (IMS)*, dirilis pada tahun 1967.

Pada tahun 1980 model relasional database menjadi paradigma DBMS yang paling dominasi. Bahasa query SQL dikembangkan untuk basis data relasional dan distandarisasikan pada akhir tahun 1980. Pada tahun 1990 mulai bermunculan DBMS seperti MySQL, PostgreSQL, MsSQL Server.

c. **DBMS (DataBase Management System)**

Untuk mengolah sebuah database, kita tidak dapat melakukanya secara manual apalagi jika datanya sudah ribuan. Oleh karena itu, kita membutuhkan sebuah software yang disebut DBMS (*DataBase Management System*). *Software* ini yang akan membantu kita dalam me-manage database yang kita buat. Terdapat berbagai macam DBMS yang dikategorikan sebagai berikut:

- *Relational Database Management Systems*
- *Hierarchical Database Management Systems*
- *Network Database Systems*
- *Object-Oriented Database Systems*
- *NoSQL Database Systems*

Namun, kita akan berfokus pada satu kategori database saja, yaitu RDBMS atau *Relational Database Management Systems*. Dalam kategori database ini, terdapat berbagai macam *software* yang menggunakannya, diantaranya Oracle, MySQL, SQL Server, SQLite, DB2, dan lain sebagainya.

d. **Schema Database.**

Schema Database adalah seperti gambaran rinci yang menjelaskan bagaimana data disusun dalam database relasional. Ini mencakup segala sesuatu mulai dari nama tabel, kolom, jenis data, hingga batasan logis lainnya. Schema database berfungsi sebagai panduan visual yang membantu kita berkomunikasi dengan arsitektur database. Dengan menggunakan schema ini, kita dapat memahami dengan jelas bagaimana setiap bagian data diorganisir dan berhubungan satu sama lain dalam database.

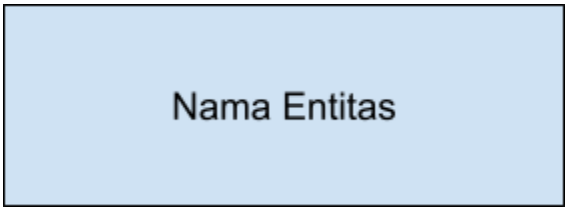
Users	
ID	Primary Key
Name	Varchar (255)
Number_Phone	Varchar (20)
Password	Text
Email	Varchar (255)

Contoh schema database.

e. **Apa itu ERD?**

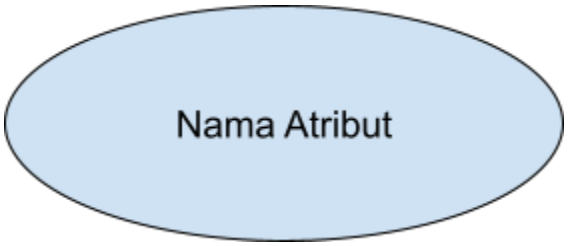
Entity Relationship Diagram adalah sebuah model penyajian data dengan menggunakan *Entity* dan *Relationship*. ERD menggambarkan model konseptual untuk menggambarkan struktur logis dari basis data dalam bentuk grafis. Tujuan dari penyajian ini adalah agar desain database dapat dipahami dan dicancang dengan mudah. Dalam ERD terdapat berbagai macam simbol yang akan kita gunakan, antara lain:

- Entitas merupakan representasi dari sebuah objek yang dapat didefinisikan dan diidentifikasikan dengan jelas. Entitas mewakili objek yang akan disimpan dalam database, Contoh seperti makhluk hidup, benda, konsep, tempat dan sebagainya yang dapat berinteraksi dengan entitas lainnya. Entitas akan direpresentasikan sebagai Tabel dalam database kita nantinya. Entitas dilambangkan dengan simbol persegi panjang dengan nama entitas di dalamnya.



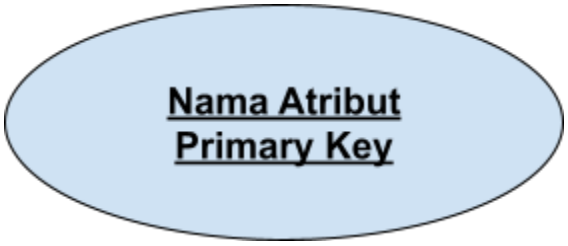
Simbol Entitas dalam ERD

- Atribut: merupakan data yang dibutuhkan sebuah entitas untuk disimpan. Atribut dapat menghubungkan antara satu entitas dan entitas lain. Sebuah atribut dapat diturunkan menjadi atribut lain yang biasa disebut atribut turunan. Sebuah atribut dilambangkan dengan simbol oval yang terhubung garis pada entitas yang memilikinya.



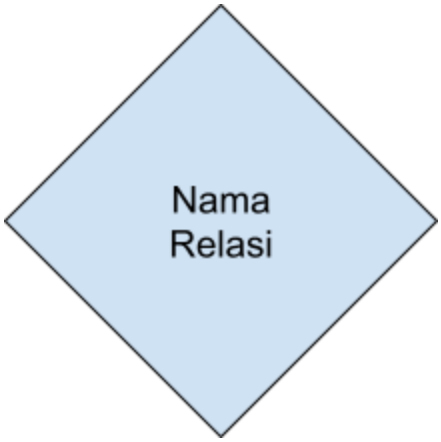
Simbol Atribut dalam ERD

- Primary Key: merupakan data yang bersifat unique yang akan membedakan satu record dengan record lainnya dalam sebuah tabel. Primary Key akan terhubung dengan foreign key pada entitas lain, sehingga kedua entitas tersebut akan saling terhubung membuat sebuah relasi. Primary Key tidak jauh berbeda dengan atribut pada umumnya, disimbolkan dengan bentuk oval akan tetapi memiliki nama atribut tercetak tebal dan bergaris bawah untuk membedakannya dengan atribut yang lain.



Simbol Atribut Primary Key dalam ERD

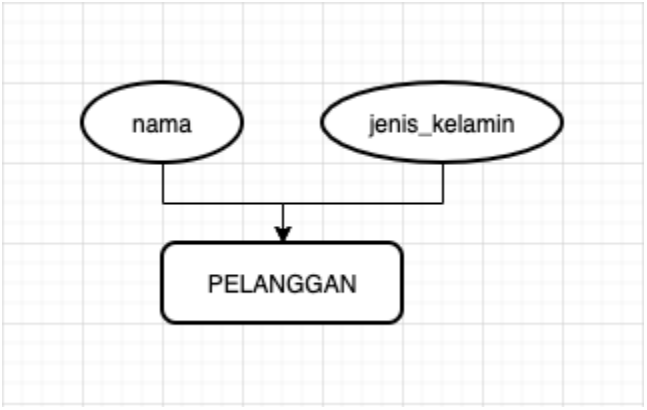
- Relasi: merupakan hubungan yang terjadi antar entitas dalam sebuah database. Relasi antar sebuah entitas memiliki beberapa jenis, yaitu: *One to One*, *One to Many*, dan *Many to Many*. Pada umumnya, relasi dilambangkan dengan simbol belah ketupat.



Simbol Relasi Antar Entitas Dalam ERD

f. **Jenis-jenis Atribut**

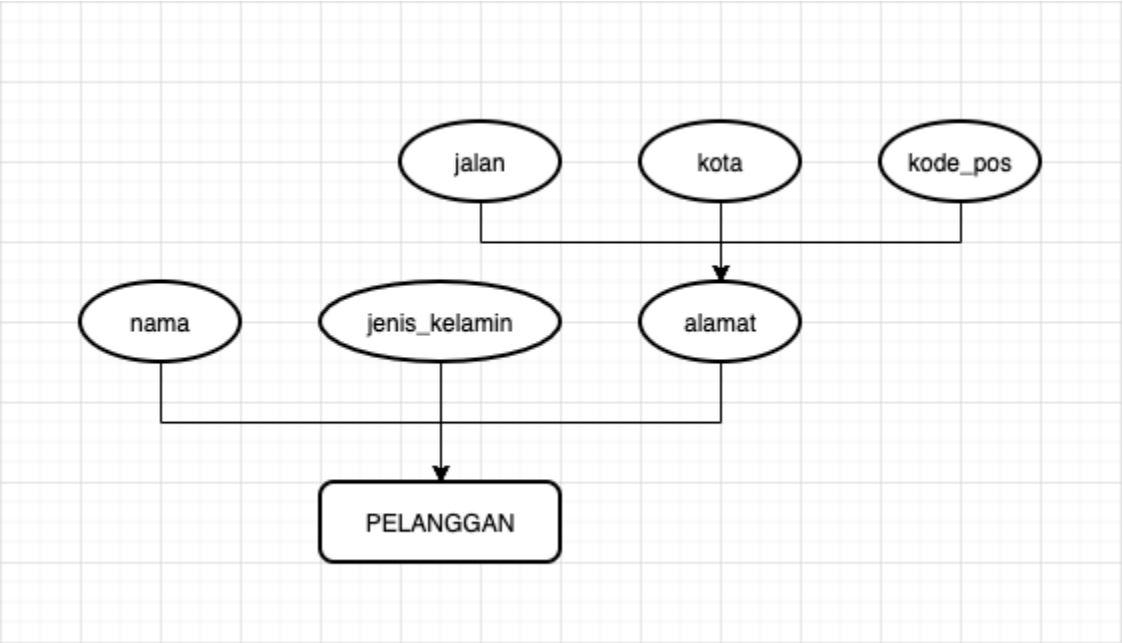
1. Atribut Sederhana (Simple) dan Komposit (Composite)
   - Atribut Sederhana (Simple/Atomik): Atribut yang terdiri atas komponen tunggal yang tidak dapat dibagi menjadi komponen yang lebih kecil.
     Contoh: atribut nama dan jenis_kelamin pada entitas pelanggan



Atribut Sederhana pada Entitas Pelanggan

   - Atribut Komposit (*Composite*): Atribut yang dapat dibagi lagi dalam beberapa bagian. Atribut yang terdiri atas beberapa komponen independen (dapat berdiri sendiri).

Contoh: Atribut alamat pada entitas pelanggan, misalnya berisi Jalan Veteran no 8, Malang, 65145. Atribut ini dapat dibagi menjadi 3 atribut, yaitu jalan (Jalan Veteran no 8), kota (Malang), dan kode_pos (65145).



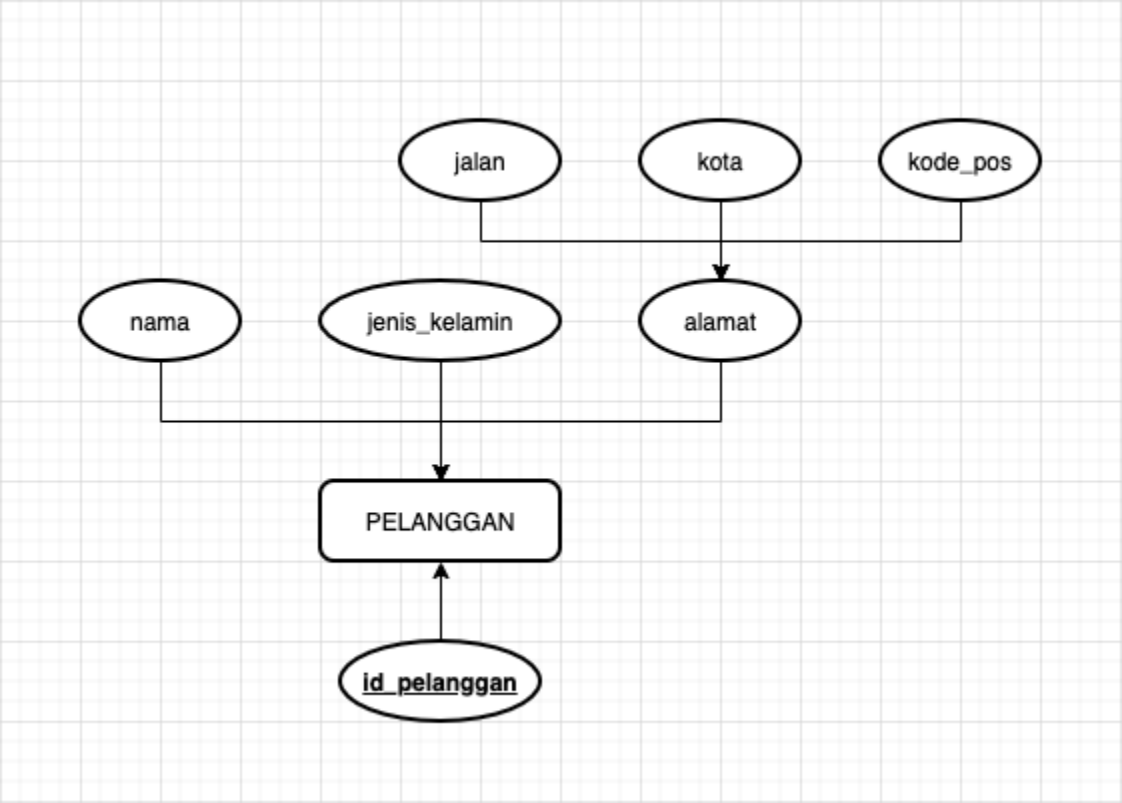
Atribut Komposit pada Entitas Pelanggan

2. Single-valued dan multi-valued attributes
   - *Single-valued* (tunggal) *Attributes*: Atribut yang memiliki paling banyak satu nilai untuk setiap baris data.
     Contoh: atribut nama pada entitas pelanggan. Setiap pelanggan hanya punya 1 nama

   - *Multi-valued Attributes*: atribut yang dapat diisi dengan lebih satu nilai tetapi jenisnya sama.
     Contoh: atribut nomor_telepon pada entitas pelanggan. Pelanggan bisa mempunyai nomor telepon lebih dari 1

3. Atribut Turunan (*Derived*): Atribut yang diperoleh dari pengolahan dari atribut lain yang berhubungan. Contoh: atribut masa studi yang diperoleh dari tanggal_lulus dikurangi tanggal_daftar

4. Primary Key
   - *Primary Key* adalah atribut yang mengidentifikasi entitas secara unik. Setiap strong entity/entitas kuat harus mempunyai Primary Key.
   - *Primary Key* bisa berasal dari satu atau gabungan dari beberapa atribut
     Contoh: Primary Key pada entitas kendaraan adalah no_stnk



Atribut Primary Key dalam Entitas Pelanggan

g. **Relasi**

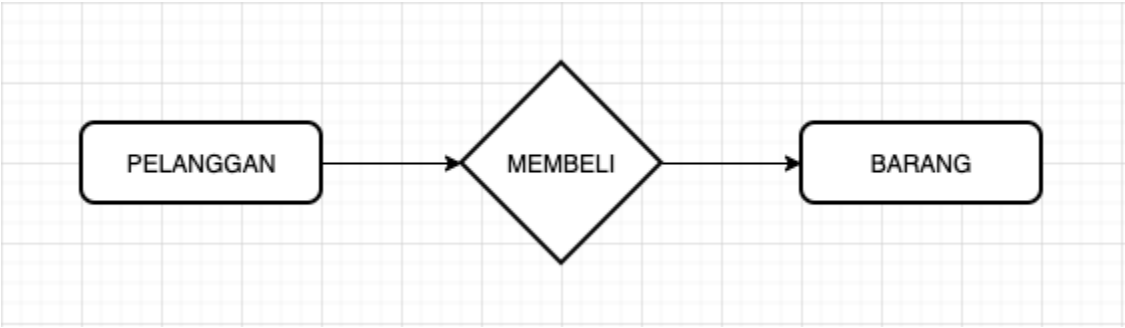
Relasi adalah hubungan antar entitas yang dapat diidentifikasi dan bermakna. Relasi dinyatakan dengan nama yang menunjukkan fungsinya. Contoh: memiliki yang menghubungkan entitas Pelanggan dan Kendaraan. Relasi dimungkinkan memiliki

atribut. Relasi digambarkan dalam bentuk belah ketupat (diamond) dengan label yang menunjukkan nama relasi, umumnya berupa kata kerja.

1. **Jenis-Jenis Relationship**

- Relasi Biner (*Binary Relationship*)

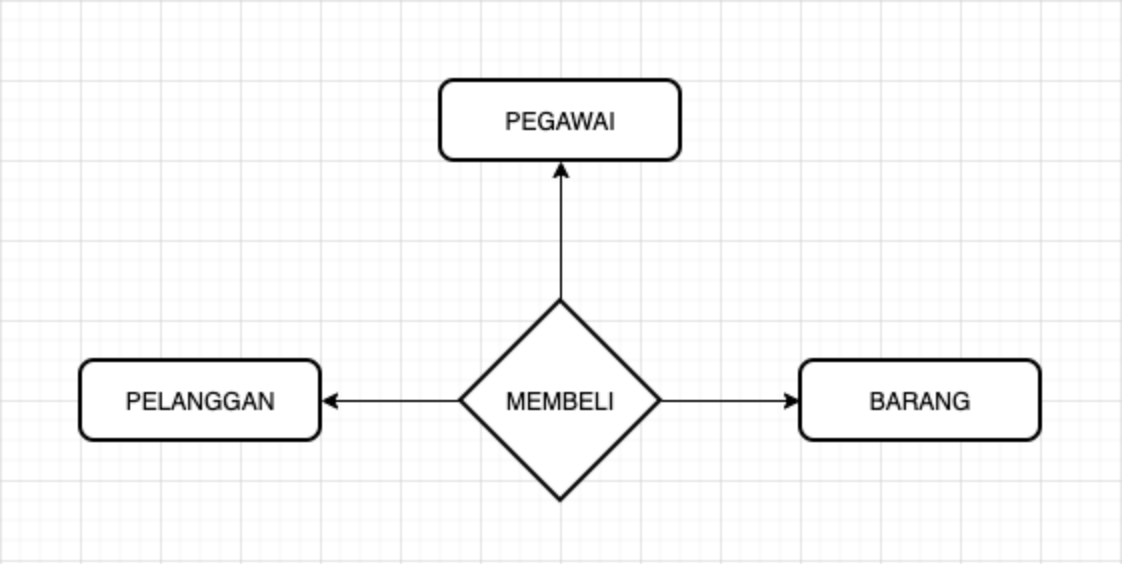
Relasi berderajat dua dinamakan relasi biner (*binary*), yakni relasi yang melibatkan dua himpunan entitas. Secara umum himpunan relasi dalam sistem basis data adalah biner.



Contoh Representasi Relasi Biner

- Relasi Ternier (*Ternary Relationship*)

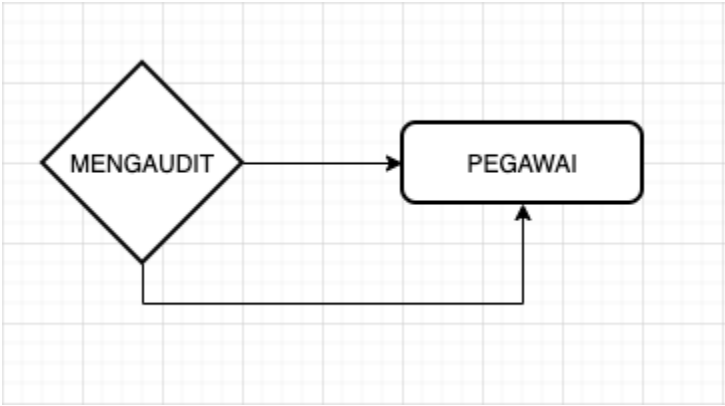
Relasi berderajat tiga dinamakan relasi ternier (*ternary*). Relasi antara lebih dari dua entitas jarang terjadi. Terdapat tiga entitas yang berpartisipasi dalam relasi ternier. Contoh: Relasi transaksi antara pelanggan yang membeli barang, dimana setiap pembelian terdapat kasir sebagai perantara yang melayani proses pembelian tersebut



Contoh Representasi Relasi Ternier

- Relasi Rekursif/Uner (Unary)

Relasi rekursif adalah tipe relasi yang menghubungkan satu entitas tunggal dengan dirinya sendiri (uner/unary). Contohnya ketika dalam sebuah badan usaha, terdapat pegawai yang berstatus sebagai manajer akan mengaudit pegawai lain yang berada di divisinya.



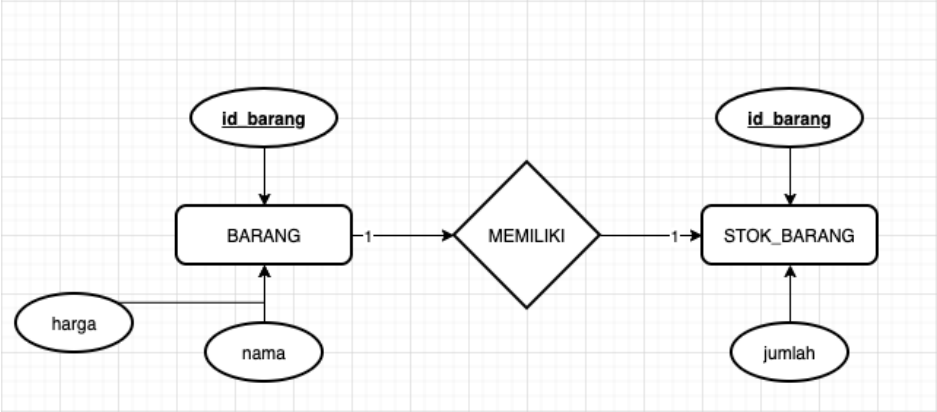
Contoh Representasi Relasi Rekursif.

2. Pemetaan Kardinalitas Relasi

Pemetaan kardinalitas relasi menggambarkan banyaknya jumlah maksimum entitas dapat berelasi dengan entitas pada himpunan entitas yang lain. Untuk Himpunan relasi biner pemetaan kardinalitasnya dapat merupakan salah satu dari tipe2 berikut :

- Satu ke Satu (*One-to-One*)

Jenis relasi ini artinya satu data dalam sebuah entitas akan terhubung dengan satu data pada entitas lain. Biasanya hubungan terjadi antara primary key pada kedua entitas.



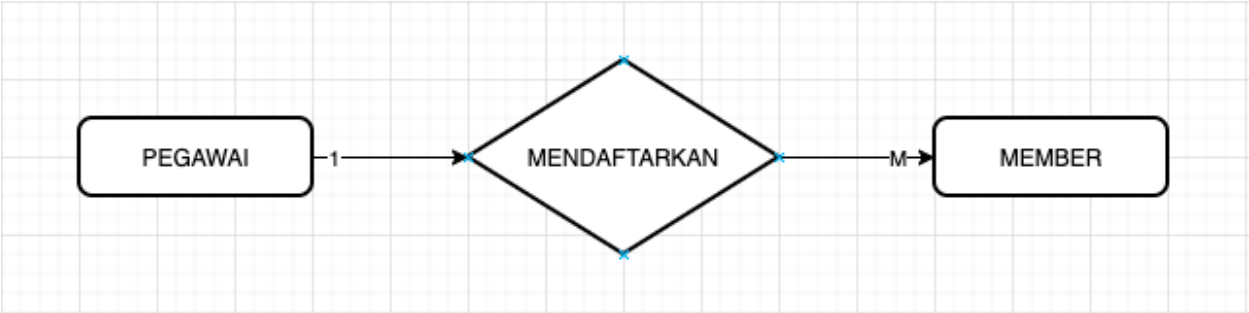
Contoh Relasi One-to-One

Seperti dapat kita perhatikan pada gambar di atas, relasi kedua entitas terjadi pada *property* id_barang. Dimana pada kedua entitas, *property* id_barang menjadi *property Primary Key* dari kedua entitas tersebut. Apabila kita jabarkan, satu data barang memiliki satu data stok_barang, sehingga bisa kita sebut relasi One-to-one.

Relasi ini sangat jarang terjadi pada pengembangan sebuah sistem, apalagi terhadap entitas yang memiliki sedikit *property*. Akan tetapi ketika entitas memiliki banyak property, maka relasi ini dapat menjadi solusi untuk menyederhanakan *structure* entitas pada database kita.

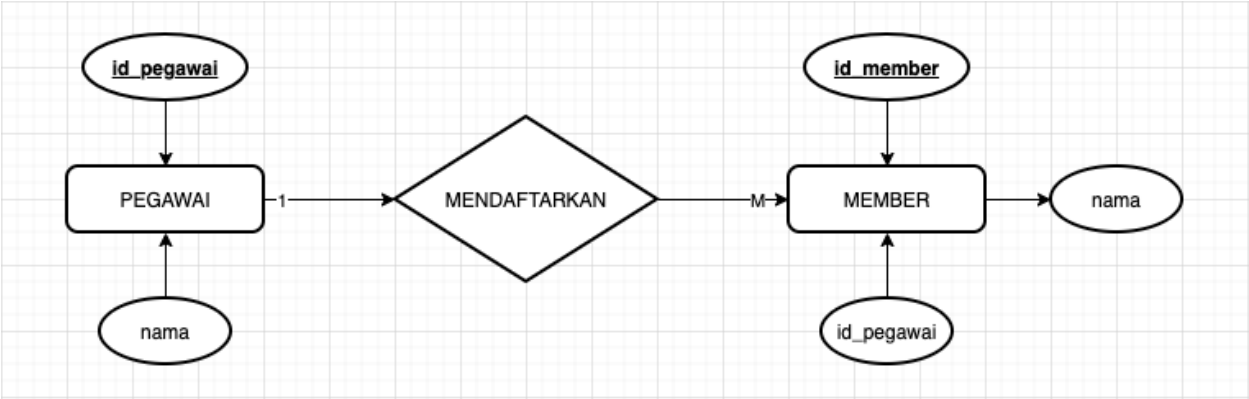
- Satu ke Banyak atau Banyak ke Satu (*One-to-Many* atau *Many-to-One*)

Relasi ini terjadi ketika satu data pada sebuah entitas, terhubung dengan banyak data pada entitas lain. Biasanya terjadi ketika *property Primary Key* dari sebuah entitas terhubung dengan *property Foreign Key* pada entitas lain.



Contoh Relasi Entitas *One-to-Many*

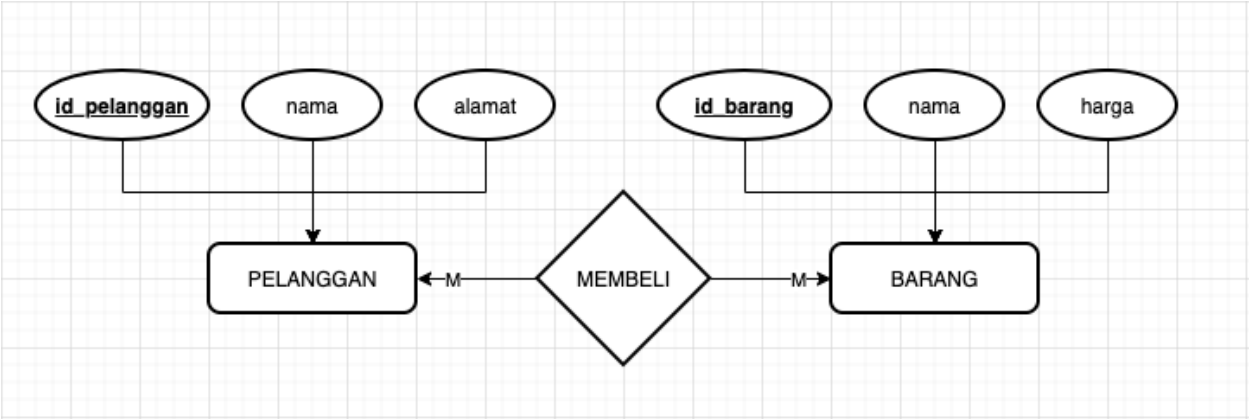
Dari gambar di atas bisa kita jabarkan bahwa, seorang pegawai dapat mendaftarkan banyak member, sedangkan seorang member hanya dapat didaftarkan oleh satu orang pegawai. Sehingga, struktur datanya terdapat sebuah *property Foreign Key* pada entitas member yang berasal dari *Primary Key* entitas pegawai. Atau dapat kita lihat pada gambar di bawah ini:



Contoh Relasi *One to Many* dengan *Foreign Key*

- Banyak ke Banyak (*Many to Many*)

Relasi ini terjadi ketika banyak data pada sebuah entitas berinteraksi dengan banyak data di entitas lain. Biasanya terjadi pada transaksi data yang berulang kedua entitas tersebut. Contohnya ketika pelanggan membeli barang, transaksi kedua entitas tersebut memungkinkan untuk terulang.



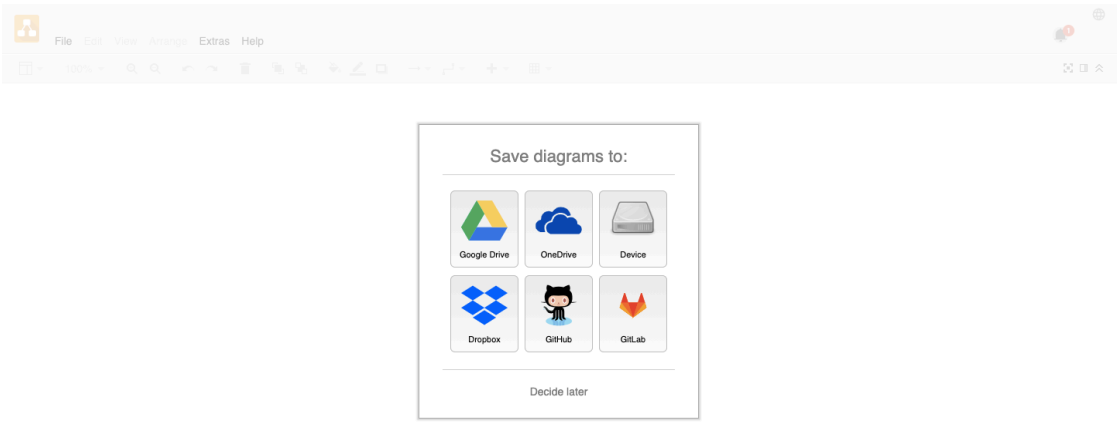
Contoh Relasi Many-to-Many

Dari gambar di atas, dapat kita jabarkan dengan kalimat seorang pelanggan dapat membeli banyak barang. Lalu, satu jenis barang dapat dibeli oleh banyak pelanggan. Sehingga, menghasilkan relasi *Many-to-Many*. Akan tetapi, relasi jenis ini **tidak diperbolehkan** terjadi dalam sebuah database, dikarenakan akan menyebabkan banyak duplikasi atau redudansi data dalam database kita. Untuk mengatasinya perlu dilakukan prosedur normalisasi yang akan kita bahas pada pembahasan selanjutnya.

D. Studi Kasus

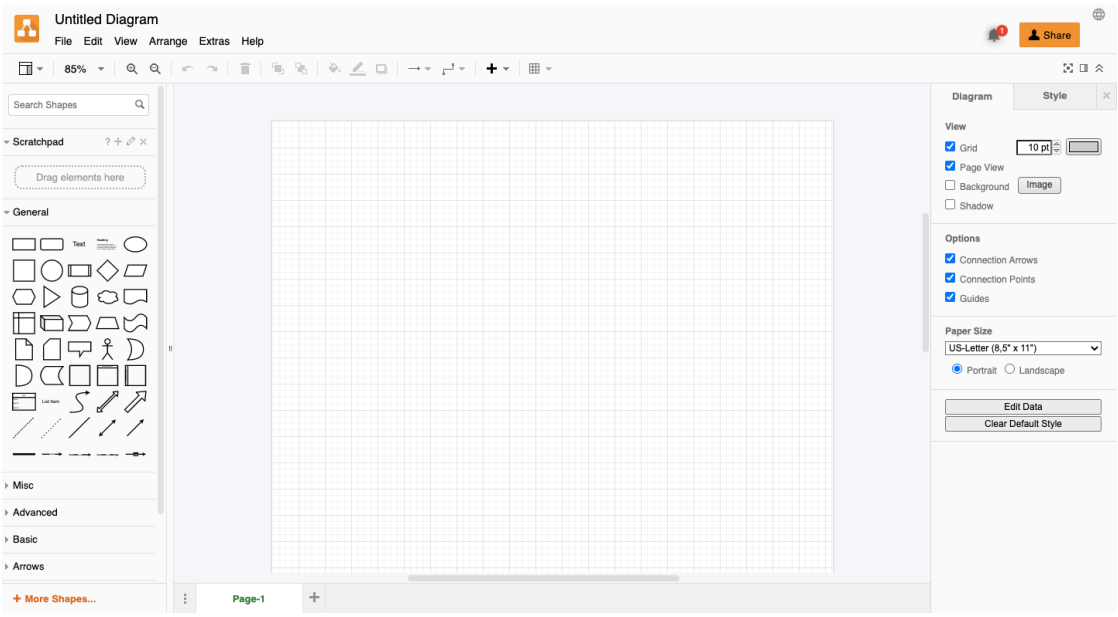
Untuk memulai merancang database kalian, kita akan menggunakan **Draw.io** yang dapat kalian akses melalui link ini. Setelah itu, ikuti petunjuk berikut ini:

1. Pilih tempat penyimpanan file diagram kalian, kalian bisa memilih untuk menyimpan file pada komputer kalian dengan memilih pilihan Device. Atau jika kalian ingin menggunakan cloud, kalian dapat memilih Google Drive.



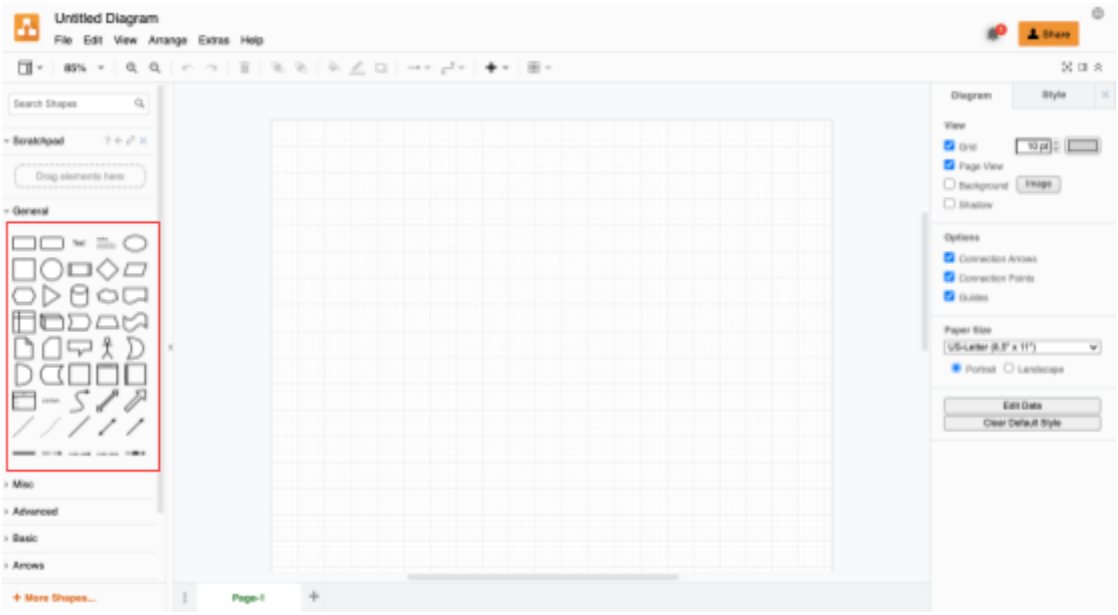
Pilih Tempat Penyimpanan File

Setelah itu akan muncul workspace tempat kalian mendesain ERD kalian.



Workspace Draw.io

2. Pada tampilan sebelah kiri, terdapat panel shape yang tersedia dalam **Draw.io**. Kita cukup menggunakan shape yang ada pada panel General.



Letak Panel General Shape

3. Selanjutnya, kalian dapat memulai menambahkan shape yang dibutuhkan untuk membuat sebuah ERD.
4. Untuk menyimpan diagram yang telah kalian buat, kalian bisa menekan menu File, lalu pilih Save as.
5. Jika muncul tampilan seperti ini, isi nama file kalian. Lalu pilih tempat penyimpanan File kalian. Misalnya kalian ingin menyimpan file kalian pada komputer atau laptop kalian. Kalian dapat memilih Device.
6. Lalu pilih folder tempat kalian ingin menyimpan file .drawio kalian lalu tekan save.

Setelah kalian siap dengan *software* **Draw.io**, mari kita coba mulai membuat ERD pertama kita. Sebagai latihan mari kita buat ERD dengan kebutuhan sebagai berikut:

1. Merk, memiliki property: id_merk, nama
2. Produk, memiliki property: id_produk, id_merk, nama, dan harga

Dari kebutuhan di atas, buatlah ERD-nya. Jangan lupa tentukan jenis relasi dari kedua entitas tersebut.

E. Referensi

Charles yanses,Apa itu DBMS, Sejarah, Manfaat, komponen, dan fungsi. https://tanyadigital.com/database-management-system-dbms-adalah/

What is a database schema? https://www.ibm.com/topics/database-schema