# Types of keys

* Primary Key (PK)
* Foreign Key (FK)
* Candidate Key
* Alternate Key
* Composite Key
* Super Key
* Unique Key
* Surrogate Key

## Primary Key (PK)

A primary key is a key used in the database to uniquely identify a record in a database. This must not be NULL values and a table can only have one primary key .This should be a unique index for a record. This can be in a two types.

* Single-column Primary Key

If there is a unique index that can be used to identify a record in a database that fields can be used as a primary key .

Here’s How to Implement in a Database:

**CREATE TABLE Customer (**

**CustomerID INT PRIMARY KEY,**

**CustomerName VARCHAR(255),**

**Address VARCHAR(255)**

**);**

* Composite Primary Key

If there is no unique index that can be used to identify a record in a database we should create such index by combining two fields.

Here’s How to Implement in a Database:

CREATE TABLE OrderDetails (

OrderID INT,

ProductID INT,

Quantity INT,

PRIMARY KEY (OrderID, ProductID)

);

## Foreign Key

A field that uniquely identify a row in another table. this can have duplicate values NULL values as such. Here simple idea is to direct to a another table from a one table

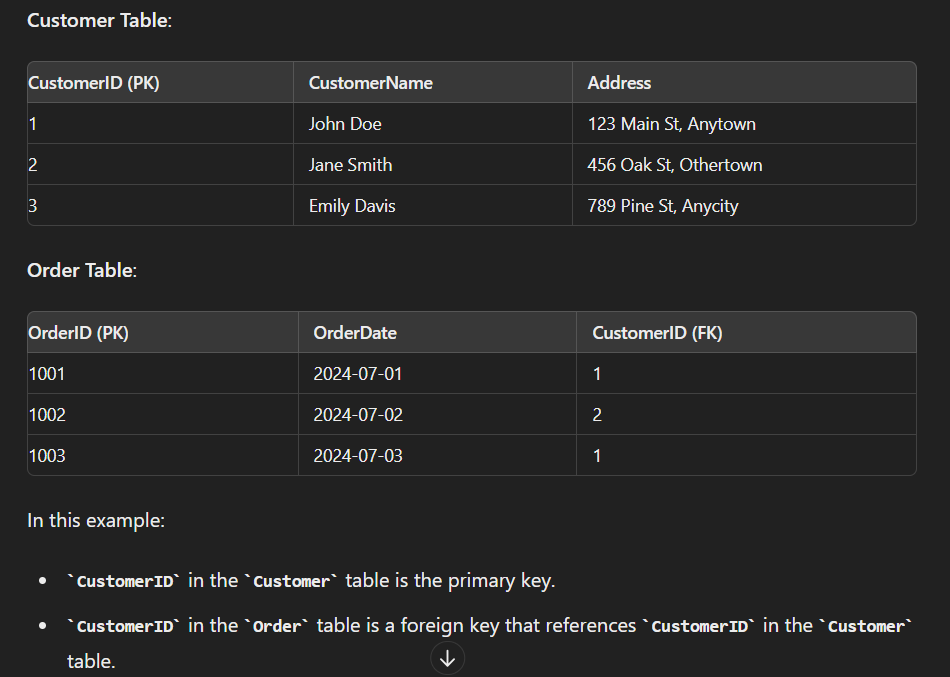
EXAMPLE:

Figure :Foreign Key concept

Here’s how to implement it in a Sql database:

**CREATE TABLE Customer (**

**CREATE TABLE Orders (**

**OrderID INT PRIMARY KEY,**

**OrderDate DATE,**

**CustomerID INT,**

**FOREIGN KEY (CustomerID) REFERENCES Customer(CustomerID)//here direct to customer table**

**);**

**CustomerID INT PRIMARY KEY,**

**CustomerName VARCHAR(255),**

**Address VARCHAR(255)**

**);**

Foreign key is important to make connection between tables to implement functions such:

* On delete Cascade(if a customer is deleted all records will be deleted)

CREATE TABLE Orders (

OrderID INT PRIMARY KEY,

OrderDate DATE,

CustomerID INT,

FOREIGN KEY (CustomerID) REFERENCES Customer(CustomerID)

ON DELETE CASCADE

);

* On Update Cascade(if customer id Updated in other tables also customer id is Updated)

CREATE TABLE Orders (

OrderID INT PRIMARY KEY,

OrderDate DATE,

CustomerID INT,

FOREIGN KEY (CustomerID) REFERENCES Customer(CustomerID)

ON UPDATE CASCADE

);

## Candidate Key

This is when there are more than one suitable candidates for Primary key one is choosen as primary key and others are . identified as a candidate keys.

Implementing:

CREATE TABLE Employee (

EmployeeID INT NOT NULL,

NationalInsuranceNumber VARCHAR(10) NOT NULL,

Email VARCHAR(255) NOT NULL,

Name VARCHAR(255),

PRIMARY KEY (EmployeeID),

UNIQUE (NationalInsuranceNumber),

UNIQUE (Email)

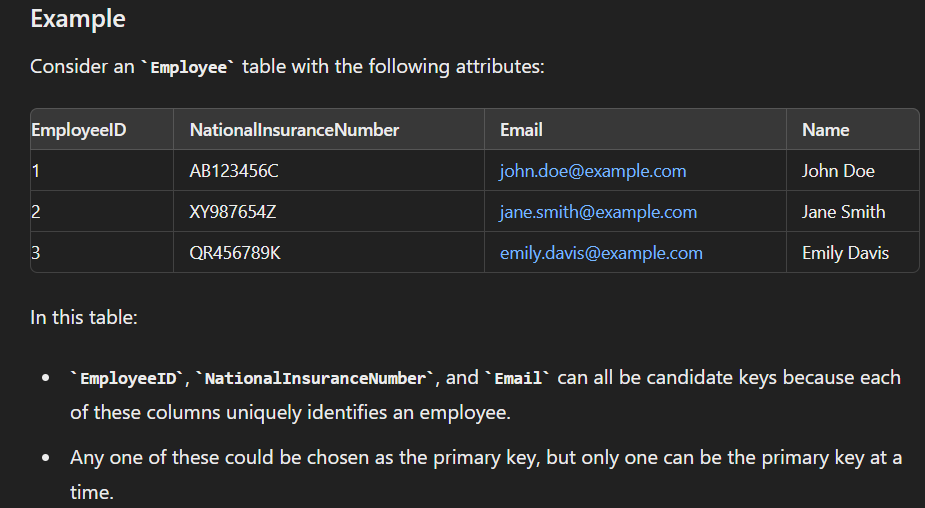
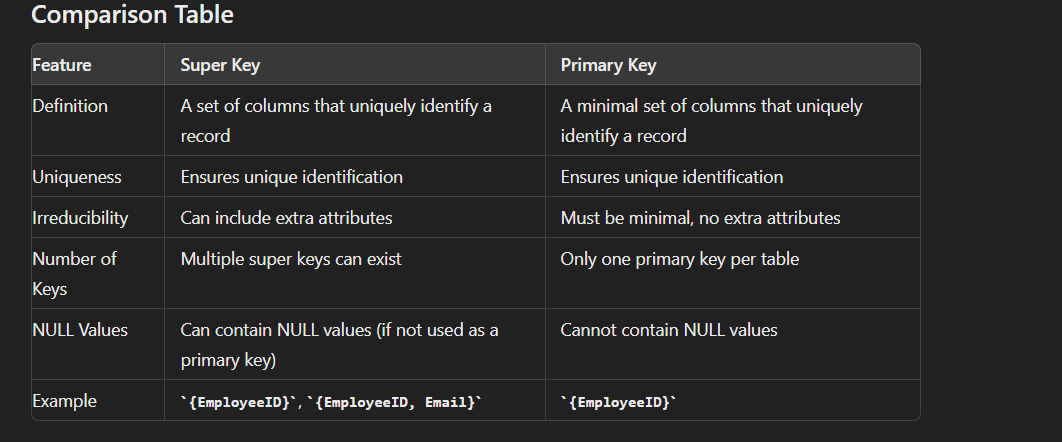
);

Figure : Candidate key example

## Super Key

One or more attributes together to form unique identifier. Does not need to be minimal.



# ER Diagram (Entity Relation diagram)

* A diagram is a drawing or a diagram that help to understand a database.
* It shows the entities and relationships of the database.
* Helps to design and implement the database.

## Terminology

### Entity: These are objects or things in the system that have a distinct existence

* + Represented by an rectangle
  + Examples include Customer, Order, Product, etc.
  + Usually name in single word

Figure :Strong entity

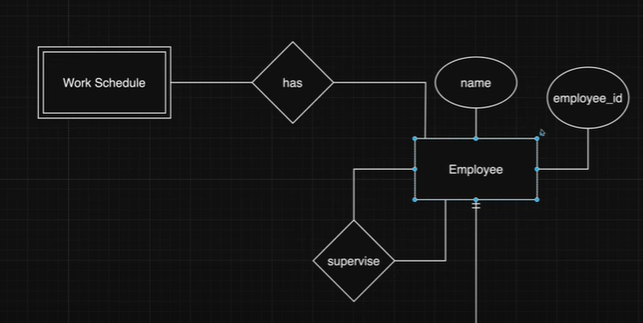
* + There are weak entities represented by double lined entity.
  + Weak entities cannot exist without a another entity .

Figure :Weak entity Vs Strong entity Behavior

### Attributes: These are the properties or details of an entity.

* + Represented by ovals and are connected to their respective entities
  + example, an Employee entity might have attributes like EmployeeID, Name, DateOfBirth, etc.
  + in above example ovals are the attributes

#### key attribute

* this is the attribute that is the primary key field
* represented by underlining the name of attribute

#### Multi valued attributes

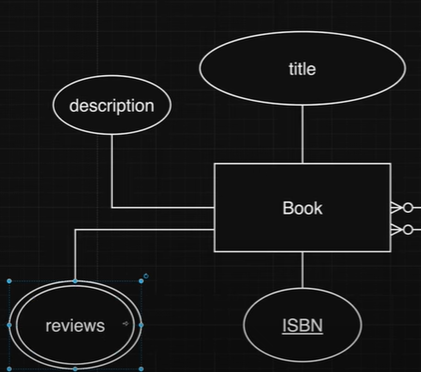
* there can be more than one value for one attribute.
* Represented with double lined ovel.
* as example there can be more than one receives for a book.

Figure :Key attribute and Multi valued attributes

#### Composite attribute

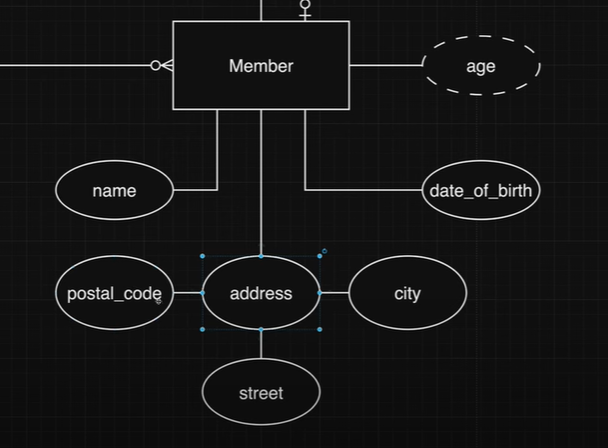
* Sometimes one attribute can be divided to more attributes and it is needed to the database
* Ex: address has city,code,district.

Figure :composite,Derived attributes

#### Derived attributes

* Represented with dashed ovel.
* This attributes can be derived by another attribute
* Ex: Age can be derived from DOB.in the above figure

## Relationships

3 Main types of relationships

### Unary

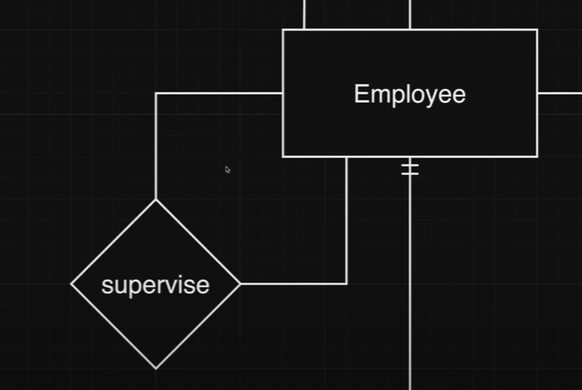
* Relationship is with only one entity to itself
* Ex : Prefect supervise student but prefect is also a student.

Figure :unary relationship

### Binary

Figure :Binary relationship

* Relationship between two entities
* Most common relationship
* EX: book and Author(author write Book)

### Ternary

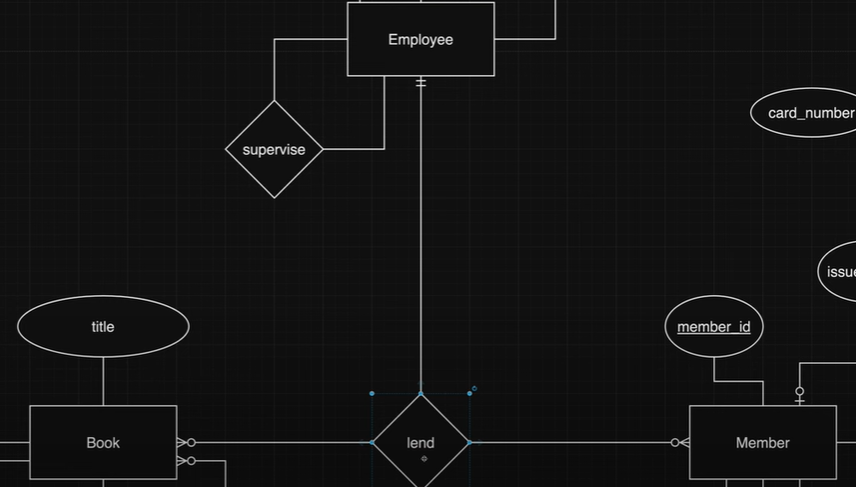
* Relationship between more than two enteties
* EX: book lend by member marked by a employee

Figure :Ternary relationship

## Cardinality

What here shows is how many entities are participate in a relationship.

### One to one

Figure :relationships One to one

### One to many

Figure :one to many and many to many

### Many to many

* Here one member can have multiple books and one book can be lend to multiple members therefore many to many relationship

# My SqL

## Creating a database

CREATE DATABASE PollyPipeDB;

USE PollyPipeDB;

## Create Tables

CREATE TABLE Customers (

CustomerID INT PRIMARY KEY,

Name VARCHAR(255),

Email VARCHAR(255),

Category VARCHAR(255)

);

## Relationships

ALTER TABLE CustomerTelephoneNumbers

ADD CONSTRAINT FK\_CustomerTelephones\_Customers

FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID);

## Normalization

Normalization is a critical process in database design that helps in organizing data efficiently, reducing redundancy, and maintaining data integrity. By following the rules of normal forms, you can create a well-structured database that supports accurate and efficient data retrieval and manipulation.

### 1N(first normal form)

* There can be any repeating groups of arrys in the table
* Must have a unique key
* Must have a primary key

### 2N(Second normal form)

* Must have be in 1N form
* There cannot be any partial dependencies. Where non-key attributes are dependent on a part of the composite primary key.

3N(Third normal form)

* Must be in 2N form
* Cannot have transitive dependency, where non-key attributes depend on other non-key attributes.

# GUI Design

## Ui design

Use figma /wireframe or powerpoint