in OOP:

- relation between classes are:

1- association 2- inheritance 3- Composition/ Aggregation

1- Association (تفاعل مباشر): no parent child relation, interaction between 2 tables/ object or more without hierarchy or oweringship (no dependency) between them.

why this is in code: To interact between 2 classes without any inheritance and no decision

(NO OWNERING SHIP).

Example: the relationship between Customer and Order or between Book and Author

**No ownership**—a book has an author, but deleting a book does not delete the author.  
  
 types: 1-1 , 1-m, m-m

2- Inheritance: interaction between 2 tables/ objects or more with specific hierarchy related to  
 specific Categorization.  
  
 why: generalization (reduce coupling and redundancy)

Types: 1-1 , 1-m, m-m

3- Composition: interaction between 2 tables/ objects or more with specific type of   
 dependencies / ownership. (one of them represent a part of the other one)

Why: to define a special relationship between 2 objects or more so if the main  
 object removed, the child one will be removed to  
 EX: Book and Pages  
  
 Types: 1-1, 1-m, or m-m

On Database:

- relation between Tables are:

\* Foreign key: special column in table that pointing to a specific row in another table why? To mark this row as a reference  
 - reference: A- owner: person own car(F) B- inheritance c- association  
Characteristics of foreign key:

1- Name: (Reference Table(b)) (Primary key of table (b))

2- Datatype: same datatype of Primary Key of table (b)

3- Constrains: foreign key (create actual reference)

4- Allow to duplicate (based on relation) {UNIQUE}

5-limitation (based on relation) {CHECK CONSTRAINT} 6- Nullable (based on Relation).

**NOTE: The location of foreign key in table is depending on the relation of the tables.**

**A) 1-m => location of foreign key in the many table**

**B) 1-1 => not necessary to put it in one and not the other (both available) but, best   
 practice IN THE SECOND TABLE.**

**C) m-m => define new table with 2 foreign keys each one of them is for one table.**

**NOTE 2: use ON DELETE CASECADE to delete child record when parent record is deleted**

**Code for STD in DB:**

**Alter Table CERTIFICATION**

**ADD StudentId int**

**ALTER TABLE CERTIFICATION**

**Add Constraint FK\_Certification\_Students\_StudentId FOREIGN KEY name (StudentId)**

**REFERENCE Students (Id)  
  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**15/02/25  
  
This is a Custom not generic because this is a PRD so everything is clear no need to brainstorming and these things.**

**Create Project To Build API using ASPnet Application To Manage Call Mangement System**

**To Service Caller and Their own calls and the system requirements should be as following**

**1- Admin 1 or more person to mange caller each one has Full name and email and password**

**2- Each Admin Add Caller each Caller Has First Name and Middle Name and Last Name and Phone Number in international format and Could add Profile Image for them and email for each one and should be unique**

**3- Each Caller May Have One or may call each call should have id and title and description and call start date and end date then flag to determine is user purchase something or not and rating from 10 for each call**

**4- admin should be able to see caller by (id, name, email, call amount, call rating)**

**5- admin should be able to see all call history for caller**

**6- admin should be able to edit caller and calls information**

**7- admin should be able to remove call from caller history but keep it on system**

**8- admin should be able to observe the overall call history (id , caller name , start , end , rate , status )**

**9- admin should be able to export calls to excel file**

**10- admin should be able to search caller by name (front end side)**

**10- admin should be able to serach calls by (phone, name, call date, isremoved from history, by rating)**