Data design

Data design in Software Engineering (SE) involves structuring and organizing the data used in the system. For creating **student notes** in an educational system, the data design should capture relevant information about students, notes, subjects, and any relationships between these entities.

1. Identify the Key Entities

- **Student**: Represents a student using the system.
- **Subject**: The subjects that students are studying.
- **Notes**: The actual notes associated with subjects and students.
- **Teacher**: Responsible for creating the notes.
- **Class**: Represents the class or batch the student belongs to.

2. Define the Attributes

Each entity will have its own set of attributes to capture the required information.

Entity: Student

- **Attributes**:
    - StudentID (Primary Key)
    - Name
    - Email
    - ClassID (Foreign Key)
    - EnrollmentDate

Entity: Subject

- **Attributes**:
    - SubjectID (Primary Key)
    - SubjectName
    - TeacherID (Foreign Key)

Entity: Notes

- **Attributes**:
    - NoteID (Primary Key)
    - Title
    - Content (Text of the note)

    - SubjectID (Foreign Key)
    - TeacherID (Foreign Key)
    - DateCreated

Entity: Teacher

- **Attributes**:
    - TeacherID (Primary Key)
    - Name
    - Email
    - Department

Entity: Class

- **Attributes**:
    - ClassID (Primary Key)
    - ClassName
    - Year

3. Define Relationships

- A **Student** belongs to a **Class**.
- A **Teacher** can create multiple **Notes**.
- A **Note** is associated with a **Subject**.
- A **Subject** is taught by a **Teacher**.
- A **Student** can access notes related to the **Subjects** they are enrolled in.

4. Create the Data Design (ER Diagram)

The following relationships are established between the entities:

- **Student** and **Class**: One-to-many (A class has many students, but a student belongs to only one class).
- **Notes** and **Subject**: One-to-many (A subject can have many notes, but each note is linked to only one subject).
- **Teacher** and **Notes**: One-to-many (A teacher creates many notes, but a note is created by one teacher).
- **Student** and **Notes**: Many-to-many (Students can access multiple notes, and notes can be accessed by multiple students).

Example Data Design in Tabular Format

Entity	Attributes
Student	StudentID (PK), Name, Email, ClassID (FK), EnrollmentDate
Class	ClassID (PK), ClassName, Year
Teacher	TeacherID (PK), Name, Email, Department
Subject	SubjectID (PK), SubjectName, TeacherID (FK)
Notes	NoteID (PK), Title, Content, SubjectID (FK), TeacherID (FK), DateCreated

6. Sample Data Records

Student Table:

StudentID	Name	Email	ClassID	EnrollmentDate
1	John Doe	john@school.com	101	2023-07-15
2	Jane Smith	jane@school.com	102	2023-07-15

Class Table:

ClassID	ClassName	Year
101	10A	2024
102	10B	2024

Teacher Table:

TeacherID	Name	Email	Department
1	Mr. Wilson	wilson@school.com	Science
2	Ms. Carter	carter@school.com	Mathematics

Subject Table:

SubjectID	SubjectName	TeacherID
1	Math	2

SubjectID	SubjectName	TeacherID
2	Science	1

Notes Table:

NoteID	Title	Content	SubjectID	TeacherID	DateCreated
1	Algebra Basics	Content here	1	2	2024-09-10
2	Physics Intro	Content here	2	1	2024-09-11

7. Use Cases

- **Student** can access all **Notes** for the subjects they are enrolled in.
- **Teacher** creates **Notes** for their subjects.
- **Admin** can manage students, subjects, and teacher assignments.

8. Normalization:

Ensure that the data is normalized to remove redundancy:

- 1st Normal Form: Ensure that the tables contain atomic values (e.g., one value per column, no repeating groups).
- 2nd Normal Form: Ensure that the table is in 1st normal form and all non-key attributes are fully functionally dependent on the primary key.
- 3rd Normal Form: Ensure that the table is in 2nd normal form and there are no transitive dependencies.