Student Management System

Phase 3: Data Modeling & Relationships

Overview

Data modeling defines the structure of your Salesforce database. Proper modeling ensures data integrity, scalability, and efficient reporting. This phase transforms business requirements into database objects and relationships.

3.1 Understanding Data Modeling Concepts

What is an Object? An object is a table in the Salesforce database. Each object contains records, and each record has fields.

Standard vs. Custom Objects:

- **Standard Objects**: Pre-built by Salesforce (Account, Contact, Opportunity)
- **Custom Objects**: Created by you for specific business needs (Student__c, Course__c, Payment__c)

3.2 Your Data Model: Objects & Fields

Object 1: Student__c (Stores Student Information)

Field Name	Field Type	Purpose
Name	Text (Unique)	Student's full name
Email	Email (Unique)	Student's email
Phone	Phone	Contact number
Date_of_Birth__c	Date	Age calculation
Total_Fees_Paid__c	Currency	Sum of all payments (updated via Apex)

Object 2: Course__c (Defines Course Offerings)

Field Name	Field Type	Purpose
Name	Text	Course name
Course_Code__c	Text (Unique)	Identifier
Fees__c	Currency	Course cost
Start_Date__c	Date	Course commencement
Status__c	Picklist	Course state

Object 3: Payment__c (Tracks Fee Transactions)

Field Name	Field Type	Purpose
Name	Auto Number	Unique payment ID
Student__c	Lookup	Links to Student
Amount_Paid__c	Currency	Payment amount
Payment_Date__c	Date	When paid
Payment_Mode__c	Picklist	How paid
Status__c	Picklist	Payment state

Object 4: Enrollment__c (Junction Object for Many-to-Many Relationship)

Field Name	Field Type	Purpose
Name	Auto Number	Unique enrollment ID
Student__c	Lookup	Links to Student
Course__c	Lookup	Links to Course
Enrollment_Date__c	Date	When student enrolled
Status__c	Picklist	Enrollment state

3.3 Implementing Relationships in Salesforce

Step 1: Create Lookup Field (One-to-Many)

Example: Add Student lookup to Payment object

1. Go to Object Manager → Payment__c → Fields & Relationships → New
2. Field Type: Lookup Relationship
3. Related To: Student__c
4. Field Label: Student
5. Field Name: Student__c
6. Relationship Name: Payments (reverse lookup)
7. Save

Result: Payment records now link to students; students show related payments list

Step 2: Create Junction Object (Many-to-Many)

Example: Enrollment__c object

1. Go to Object Manager → Create → Custom Object
2. Label: Enrollment
3. Plural Label: Enrollments
4. Object Name: Enrollment
5. Create two Lookup fields:
    - Lookup 1: Student__c (related to Student)
    - Lookup 2: Course__c (related to Course)
6. Save

Result: Supports flexible student-course combinations

3.4 Advanced Modeling: Validation Rule

Goal: Prevent overpayment (payment amount exceeds course fee)

Formula Logic:

(Student__r.Total_Fees_Paid__c + Amount_Paid__c) > Student__r.Course_Fee__c

Implementation:

1. Go to Object Manager → Payment__c → Validation Rules → New
2. Rule Name: Prevent_Overpayment
3. Formula: Above formula
4. Error Message: "Total payment exceeds the course fee. Please check the amount."
5. Error Location: Amount_Paid__c field
6. Save & Activate

Result: System prevents saving payment if total would exceed course fee