# Phase 3: Data Modelling & Relationships

**1. Standard & Custom Objects**

For CauseConnect CRM, I designed a simple data model using a mix of standard objects (already in Salesforce) and custom objects.

Standard Objects Used

- Contact: represents donors (individuals)
- Campaign: represents fundraising campaigns (optional, but useful to track donation sources)
- User: represents internal users like Fundraising Manager and Program Manager

Custom Objects Created

- Program: represents NGO programs or initiatives
- Beneficiary: represents people/communities supported by programs
- Donation: represents donations made by donors to programs

Standard objects let me use Salesforce's built-in features (contacts, reports, campaigns), while custom objects give me full control over NGO-specific data that Salesforce doesn't provide by default.

**2. Fields**

I created specific fields for each custom object to capture the data needed for NGO operations. Below is the current field structure I configured:

**Program Object**

| **Field Label** | **API Name** | **Type** | **Notes** |
|---|---|---|---|
| Program Name | Name | Text (standard) | Primary name field |
| Program Start Date | Program_Start_Date__c | Date | When the program started |
| Program End Date | Program_End_Date__c | Date | Optional, when it ended |
| Program Budget | Program_Budget__c | Currency | Estimated budget |

| Field Label | API Name | Type | Notes |
|---|---|---|---|
| Description | Description__c | Long Text | About the program |

**Beneficiary Object**

| Field Label | API Name | Type | Notes |
|---|---|---|---|
| Beneficiary Name | Name | Text (standard) | Primary name field |
| Age | Age__c | Number | Optional demographic info |
| Gender | Gender__c | Picklist (Male, Female, Other) | -- |
| Contact Number | Contact_Number__c | Phone | -- |
| Program | Program__c | Lookup (Program) | Links each beneficiary to a program |

**Donation Object**

| Field Label | API Name | Type | Notes |
|---|---|---|---|
| Donation Name | Name | Text (standard) | Primary name field |
| Amount | Amount__c | Currency | Amount donated |
| Date | Donation_Date__c | Date | When donation was made |
| Donor (Contact) | Donor__c | Lookup (Contact) | Links donation to donor |
| Program | Program__c | Lookup (Program) | Shows which program funds are used for |
| Frequency | Frequency__c | Picklist | Frequency of recurring donations |

**Field-Level Security**

- While creating fields, I made them **Visible (not Read-Only)** for the main internal profiles: Fundraising Manager, Program Manager, Field Officer.
- I will later restrict field-level access for external users (Donors, Auditors).

**3. Record Types**

I used record types to manage different kinds of records within the same object. This allows me to tailor page layouts and processes for each type.

1. Donation Object
    a. One-Time Donation for single contributions
    b. Recurring Donation for ongoing pledges

*I created two record types under the Donation object. This way, when a user creates a new donation record, they can choose whether it's a one-time donation or a recurring one. In the future, I can add automation or reports that specifically analyse recurring donations separately from one-time contributions.*

2. Beneficiary Object

*Initially, I considered using record types for Individual vs Group beneficiaries. After reviewing the requirements, I decided to keep things simple and stick to a single record type. This avoids unnecessary complexity while still capturing essential data.*

3. Program Object

*For now, only one record type is used for programs. If the NGO expands into different categories of programs (e.g., Health vs Education), I can introduce additional record types in the future.*

**4. Page Layouts**

For each object, I customized the page layouts to display the most relevant fields in a clean order.

• Program Layout: shows program details (start/end date, budget, description).

• Beneficiary Layout: includes demographic details, contact information, and program lookup.

• Donation Layouts: one layout for one-time donations and another for recurring donations. These layouts show donor, amount, date, payment status, and related program.

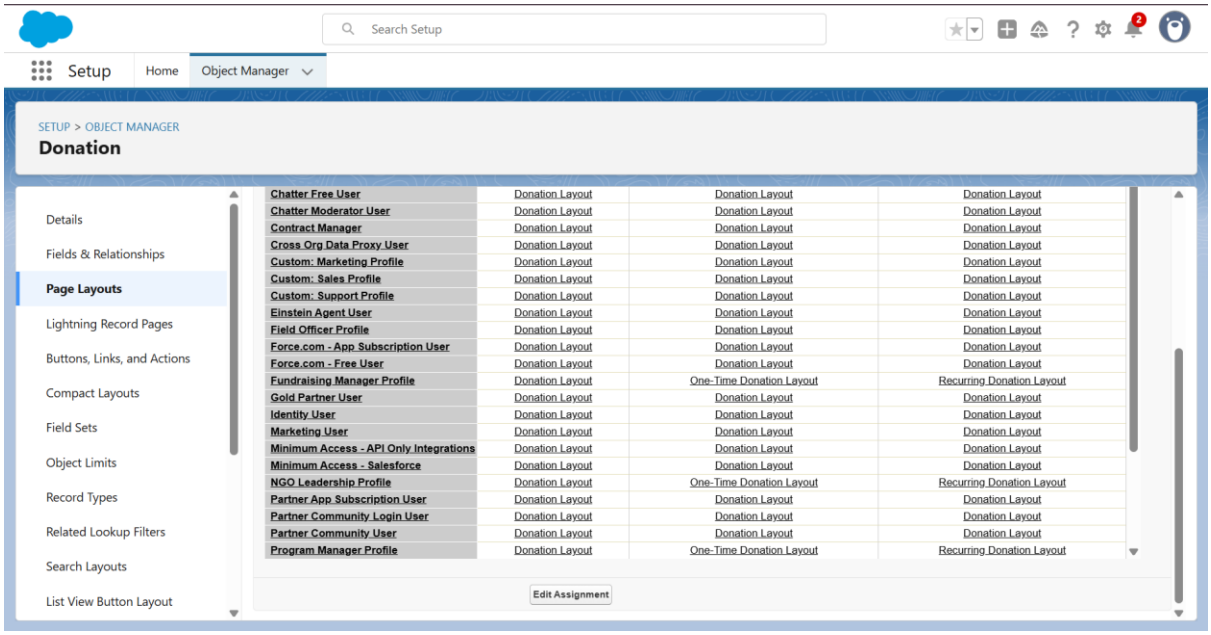


*Figure 1:Page Layouts*

### 5. Compact Layouts

I configured compact layouts so that important information is visible at the top of each record.

- Program Compact Layout:
    - Program Name
    - Start Date
    - End Date
    - Status
    - Owner
    - Budget
- Beneficiary Compact Layout:
    - Beneficiary Name
    - Age
    - Gender
    - Program
    - Contact Number
- Donation Compact Layout:
    - Donation Name
    - Donor (Contact)
    - Amount
    - Donation Date
    - Payment Status

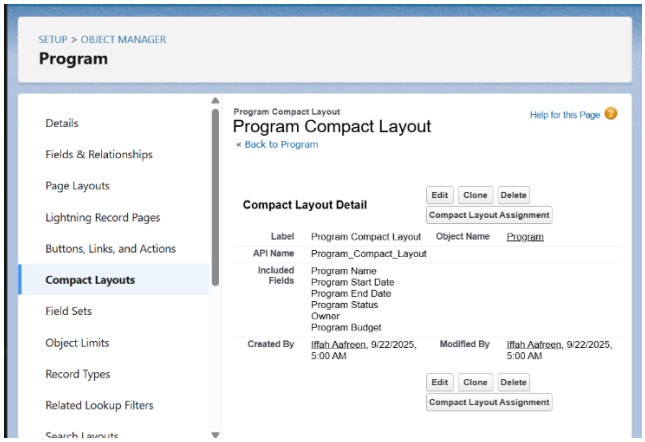


*Figure 2: Program Compact Layout. Similarly made for Beneficiary and Donation*

## 6. Schema Builder

I used Schema Builder in Salesforce to visualize the relationships between the objects.

Relationships:

• Program → Beneficiary (Lookup): A program can have many beneficiaries.

• Program → Donation (Lookup): A program can have many donations.

• Donation → Contact (Lookup): A donor (contact) can be linked to many donations.

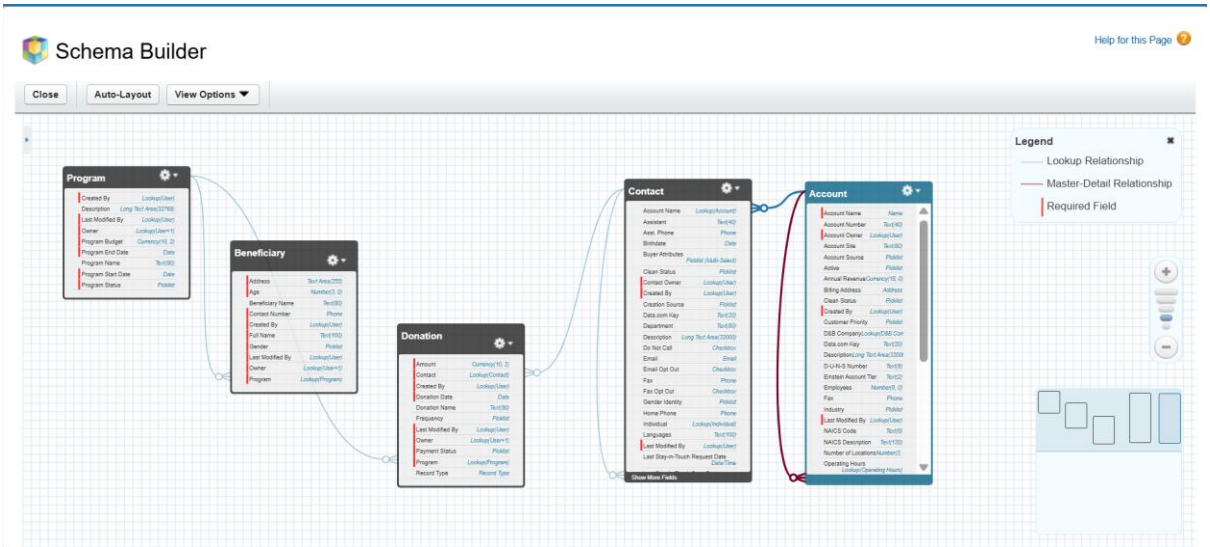


*Figure 3:Schema View*

## 7. Relationship Types

I chose Lookup relationships instead of Master-Detail for flexibility.

• Lookup allows records to exist independently (e.g., a donation can exist even without being linked to a program).

• Master-Detail would have forced every child record to be dependent on the parent, which didn't fit all use cases.

• Hierarchical relationships were not relevant since they apply only to the User object.

**8. Junction Objects**

At this stage, I did not implement junction objects because the current requirements did not involve many-to-many relationships. However, I documented a potential future use case:

> *If one Beneficiary can belong to multiple Programs, and one Program can support multiple Beneficiaries, then a junction object like "Program Beneficiary" would be needed.*

**9. External Objects**

I did not create external objects in this phase. However, I noted their possible future use.

> *If donation payment data is stored in external systems like Stripe or PayPal, external objects could be used to surface that data inside Salesforce without storing it directly.*

This makes the system more scalable and better integrated with third-party services.