# Phase 2: Data Modeling & Custom Objects (Migrant Health CRM)

## Objective

The objective of Phase 2 is to design the data structure for the Migrant CRM system. In this phase, we create all the custom objects, fields, relationships, and navigation tabs required to store and organize patient-related information.  
  
This data model will serve as the foundation for all further phases, such as automation, triggers, Lightning Web Components (LWC), reporting, and analytics.

## Use Case

The Migrant CRM is designed for NGOs and healthcare organizations to manage patients, their appointments, and migrant follow-ups in one place.  
  
- Every Patient record should store basic details like name, contact, age, gender, and case manager.  
- Each Appointment should be linked to a patient and should track date, time, status, and assigned doctor.  
- Every Migrant Record should track follow-up status after an appointment is completed.

## Step 1: Create Custom Object – Patient

Purpose / Use Case:  
We need a central object to store basic patient information like name, gender, age, contact, and who is managing them.  
  
Exact Steps:  
- Go to Setup → Object Manager → Create → Custom Object.  
- Fill the following details:  
 Label: Patient  
 Plural Label: Patients  
 Object Name (API): Patient\_\_c  
 Record Name: Patient Name (Text)  
  
Check the following options:  
✅ Allow Reports  
✅ Track Activities  
✅ Launch New Tab Wizard  
  
Click Save.

## 

## 🧪 Step 2: Add Fields to Patient Object

Use Case: We need fields to store patient details and help staff search, filter, and manage them.

|  |  |  |  |
| --- | --- | --- | --- |
| Field Label | Data Type | API Name | Description |
| Age | Number (3,0) | Age\_\_c | Patient's age |
| Gender | Picklist | Gender\_\_c | Male / Female / Other |
| Priority | Picklist | Priority\_\_c | High / Medium / Low |
| Case Manager | Lookup (User) | Case\_Manager\_\_c | Assigns a patient to a case manager |
| Phone | Phone | Phone\_\_c | Patient’s phone number |
| Email | Email | Email\_\_c | Patient’s email address |
| Patient ID | Text | Patient\_ID\_\_c | Unique ID for patient (e.g., PAT-001) |
| DOB | Date | DOB\_\_c | Patient's Date of Birth |

## 

## 🩺 Step 3: Create Custom Object – Appointment

Purpose / Use Case:  
We need to record every appointment booked for a patient and track its status.  
  
Exact Steps:  
- Go to Setup → Object Manager → Create → Custom Object → New Custom Object.  
- Fill details:  
 Label: Appointment  
 Plural Label: Appointments  
 API Name: Appointment\_\_c  
 Record Name: Appointment Number (Auto Number, Format: APPT-{0000})  
  
Check:  
✅ Allow Reports  
  
Click Save.

## WhatsApp Image 2025-09-24 at 19.31.53_b444868f

## Step 4: Add Fields to Appointment

|  |  |  |  |
| --- | --- | --- | --- |
| Field Label | Data Type | API Name | Description |
| Appointment Date | Date/Time | Appointment\_Date\_\_c | Date and time of appointment |
| Status | Picklist | Status\_\_c | Scheduled / Completed / Cancelled |
| Patient | Lookup (Patient) | Patient\_\_c | Links appointment to a patient |
| Doctor | Lookup (User) | Doctor\_\_c | Doctor assigned to the appointment |
| Reminder Sent | Checkbox | Reminder\_Sent\_\_c | Default False – used in automation later |

## 

## Step 5: Create Custom Object – Migrant Record

Purpose / Use Case:  
After an appointment is marked Completed, we must track the patient’s follow-up and recovery progress.  
  
Steps:  
- Go to Setup → Object Manager → New Custom Object.  
- Fill details:  
 Label: Migrant Record  
 Plural Label: Migrant History  
 API Name: Migrant\_Record\_Number\_\_c  
 Record Name: Migrant Record Number (Auto Number, Format: WR-{0000})  
  
✅ Allow Reports  
  
Save.

## 

## Step 6: Add Fields to Migrant Record

|  |  |  |  |
| --- | --- | --- | --- |
| Field Label | Data Type | API Name | Description |
| Patient | Lookup (Patient) | Patient\_\_c | Links to the patient record |
| Appointment | Lookup (Appointment) | Appointment\_\_c | Links to the appointment that generated record |
| Status | Picklist | Status\_\_c | Follow-up Pending / Done |
| Notes | Long Text Area | Notes\_\_c | Notes or updates from follow-up |

## 

## Step 7: Establish Object Relationships

Use Case:  
We want a complete relational data model:  
  
- One Patient → Many Appointments  
- One Appointment → One Migrant Record  
- One Patient → Many Migrant Records  
  
Relationships:  
- Patient\_\_c (Lookup) in Appointment links appointment to patient.  
- Appointment\_\_c (Lookup) in Migrant Record links it back to appointment.  
- Patient\_\_c (Lookup) in Migrant Record links to patient too.

## 

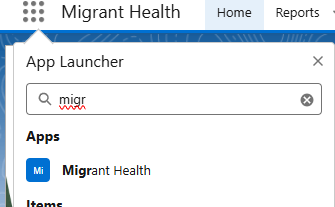
## Step 8: Create Tabs for Objects

Purpose:  
So that users can easily navigate to these records from the main navigation bar.  
  
Steps:  
- Setup → Tabs → New → Custom Object Tab.  
- Create tabs for:  
 • Patient  
 • Appointment  
 • Migrant Record

## 

## Step 9: Add Objects to Migrant Health CRM App

Steps:  
- Go to Setup → App Manager → Edit → Migrant Health CRM App.  
- Click Navigation Items → Add Tabs:  
 • Patients  
 • Appointments  
 • Migrant Records  
  
Click Save.



## ✅ Final Data Model Summary

|  |  |
| --- | --- |
| Object | Purpose |
| Patient\_\_c | Stores all patient details |
| Appointment\_\_c | Tracks appointments for patients |
| Migrant\_Record\_Number\_\_c | Tracks follow-up care and migrant progress |

Relationships:  
- Patient → Appointment (One-to-Many)  
- Appointment → Migrant Record (One-to-One or One-to-Many)  
- Patient →Migrant Record (One-to-Many)