**A close-up of a logo

Description automatically generated**

**DATABASE DESIGN FOR**

**Non-Governmental Organizations (NGOs)**

**"UNITY FOR LEBANON"(UFL)**

**by**

**KEVIN’s group**

**Rami Aabed**

[**rami.aabed@lau.edu**](mailto:rami.aabed@lau.edu)

**Imad Mahmoud**

[**imad.mahmoud@lau.edu**](mailto:imad.mahmoud@lau.edu)

**Ghady Baydoun**

[**ghady.beydoun@lau.edu**](mailto:ghady.beydoun@lau.edu)

**Kevin Abdelnour**

[**kevin.abdelnour@lau.edu**](mailto:kevin.abdelnour@lau.edu)

**Nour Al Ain Al Hajj**

[**nouralain.alhajj@lau.edu**](mailto:nouralain.alhajj@lau.edu)

**Introduction**

Non-Governmental Organizations (NGOs) are pivotal in addressing various social issues, including poverty alleviation, education, healthcare, environmental protection, and human rights advocacy. The complexity of their operations necessitates a robust database application that can efficiently manage their activities, resources, and relationships. This document outlines a proposed database application for an NGO, detailing its structure, functionalities, and the relationships between different entities involved in its operations.

In Lebanon, where communities have gone through the hardships of war and suffer from the present day, "Unity for Lebanon"(UFL) stand as a pillar of hope, providing resilience and support. Created for the purpose of providing help to people from war and resettlement, it has always been a trusted source of support to the needy in the community whom are seeking food, shelter, medical care, and psychological support.

However, in the face of violence and chaos, UFL has experienced a profound setback; the loss of crucial records and information, which are the essence of the program and community engagement. Yet this did not defeat UFL, it has made its' mission even more critical, and today it is taking measures aimed at restoring the situation.

**Overview of the NGO Database Application**

"Unity for Lebanon" (UFL) was founded in the aftermath of Lebanon's civil conflicts, a time when local communities were struggling to rebuild from destruction and displacement. Established by a group of humanitarian activists, UFL began with the simple mission of delivering food and medical supplies to families affected by the war. Over the years, the organization evolved, expanding its services to address the full spectrum of needs within vulnerable communities, including psychological support, housing, and job training programs.

Rooted in values of unity, resilience, and hope, UFL quickly earned the trust of communities throughout Lebanon. The NGO grew into a network of dedicated volunteers, health professionals, and social workers, building strong local connections and responding to emergencies as well as long-term recovery needs. UFL’s work also expanded to supporting refugees, particularly those fleeing the Syrian crisis, providing them with vital resources and helping them integrate into Lebanese society.

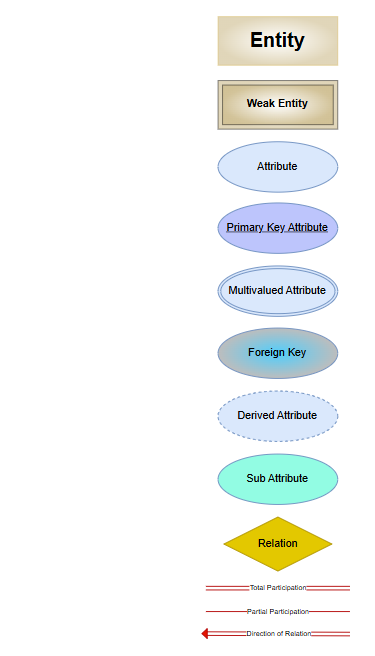
UFL’s headquarters, a modest center in the heart of Beirut, became a hub for community engagement, where people could seek help or volunteer their time. This center stored detailed records of families, resources, and local partnerships, critical data used to coordinate effective support. However, recent political unrest and social turmoil led to a tragic loss of these records, a setback that could have undermined years of community trust and operational effectiveness.

Refusing to be defeated, UFL’s leaders viewed this challenge as a call to action, doubling down on efforts to recover lost data and secure future information. Today, UFL is working to rebuild its database, develop digital safeguards, and engage with communities anew, determined to continue being a beacon of hope in Lebanon’s humanitarian landscape.

**Objectives of the Database Application**

1. **Centralized Information Management**: To provide a single platform for storing and managing all relevant data related to the NGO's operations.
2. **Enhanced Reporting**: To generate insightful reports that can help in decision-making and strategic planning.
3. **Improved Communication**: To facilitate better communication among donors, volunteers, staff, and beneficiaries.
4. **Resource Allocation**: To optimize the allocation of resources based on project needs and donor contributions.
5. **Impact Assessment**: To measure the effectiveness of projects and services provided to beneficiaries.

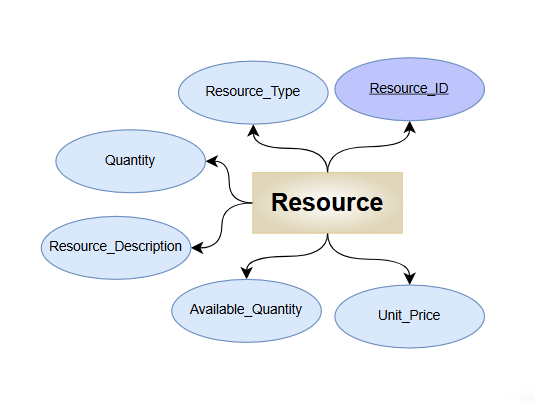
It is incredibly important to mention the ER diagram symbol annotations; therefore, we mention and annotate them here. This will help you understand our ER faster and more firmly:



**Entity Types**

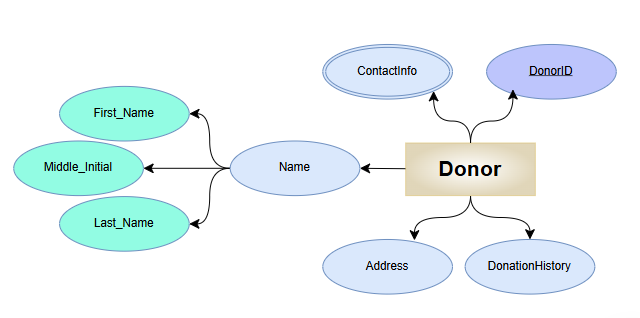
The following are the ten different entity types that will be included in the database application:

1. **Resource**

****

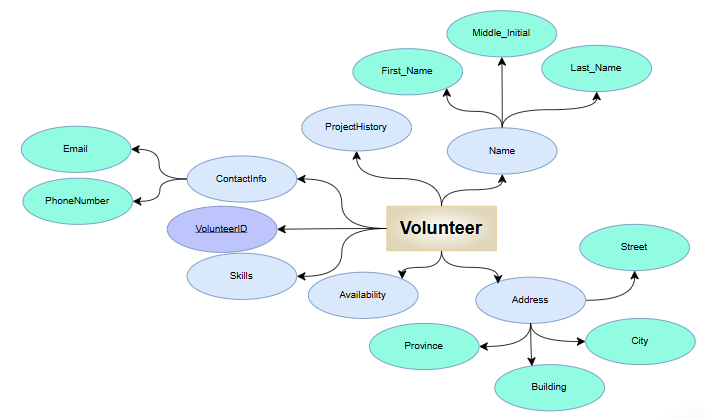
* + **Description**: The Resource entity encompasses all types of resources that the NGO utilizes in its operations. This includes human resources (staff and volunteers), financial resources (funds), and material resources (supplies and equipment). Proper management of resources is vital for the effective functioning of the NGO.
  + **Attributes**:
    - **Resource\_ID** (Primary Key): A unique identifier for each resource, allowing for precise tracking and management.
    - **Resource\_Type**: The type of resource (e.g., Human, Financial, Material), which categorizes resources for better organization and reporting.
    - **Resource\_Description**: A detailed description of the resource, providing context and specifics about its use and importance.
    - **Quantity**: The total quantity of the resource available, which helps in inventory management and planning.
    - **Available\_Quantity**: The quantity of the resource currently available for use, allowing the NGO to assess its current capacity.
    - **Unit\_Price**: The unit price of the resource (if applicable), which is essential for budgeting and financial planning.

1. **Donor**

****

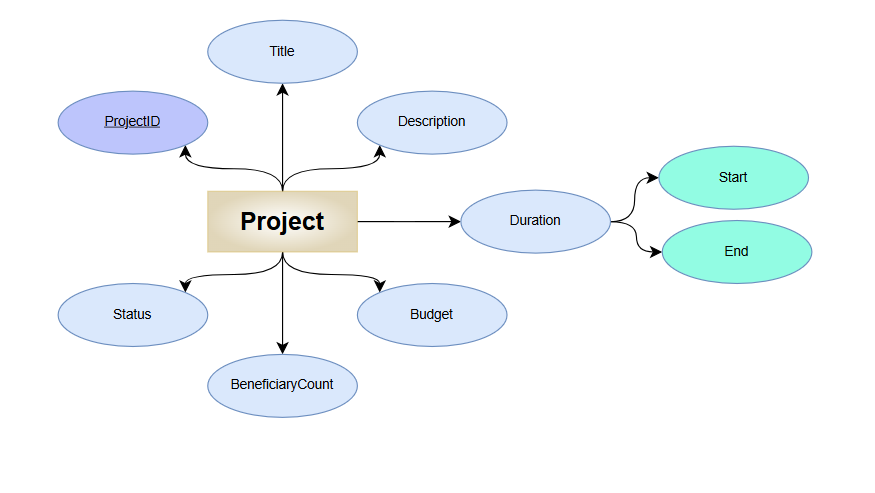
* **Description**: Donors are individuals or organizations that contribute funds or resources to the NGO. They play a crucial role in the sustainability of the NGO's operations. Donors can be categorized into various types, such as individual donors, corporate sponsors, and grant-making foundations. Understanding donor preferences and history is essential for maintaining relationships and encouraging future contributions.
* **Attributes**:
* **DonorID** (Primary Key): A unique identifier for each donor.
* **Name**: Composite attribute that consists of FirstName, MiddleName (if applicable), and LastName, allowing for a complete representation of the donor's name.
* **ContactInfo**: Multi-valued attribute that includes multiple pieces of information, such as Email and PhoneNumbers.
* **Address**: Composite attribute that includes multiple components: Street, City, Province, and Building, providing detailed location information.
* **DonationHistory**: A list of donations made by the donor, including amounts and dates.

1. **Volunteer**

****

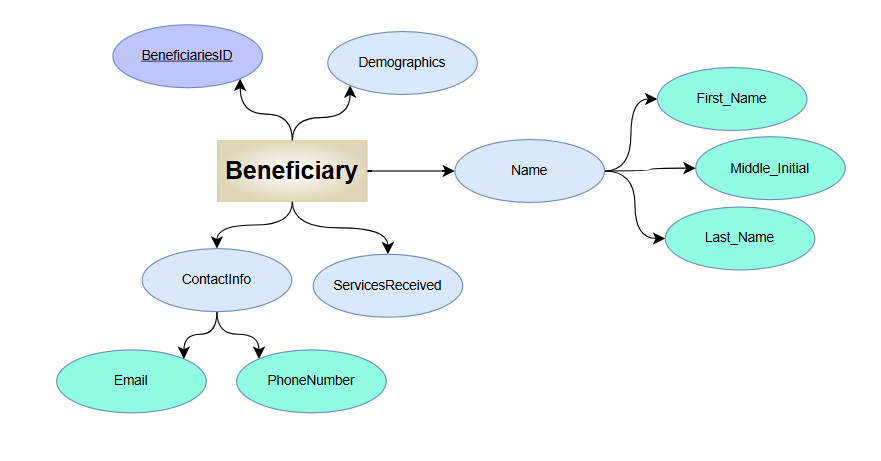
* + **Description**: Volunteers are individuals who offer their time and skills to support the NGO's activities. They are often the backbone of many NGOs, providing essential services without financial compensation. Volunteers may have specific skills that are beneficial for certain projects, such as teaching, medical assistance, or administrative support.
* **Attributes**:
* **VolunteerID** (Primary Key): A unique identifier for each volunteer.
* **Name**: Composite attribute that consists of FirstName, MiddleName (if applicable), and LastName, allowing for a complete representation of the volunteer's name.
* **ContactInfo**: Composite attribute that includes Email and PhoneNumbers, providing multiple ways to contact the volunteer.
* **Skills**: Multi-valued attribute that contains a list of skills that the volunteer possesses (e.g., teaching, event planning, fundraising).
* **Availability**: Days and times the volunteer is available to work.
* **ProjectHistory**: A multi-valued attribute that contains a list of projects the volunteer has participated in, including roles and contributions.

1. **Project**

****

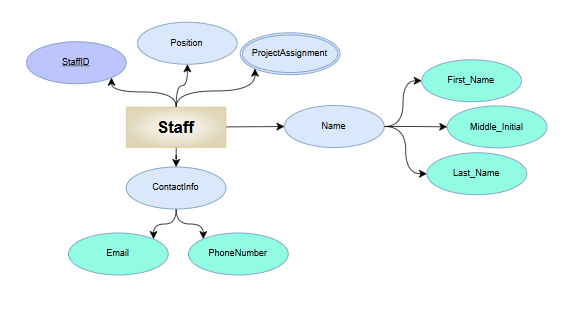
* **Description**: Projects are specific initiatives undertaken by the NGO to achieve its goals. Each project has defined objectives, timelines, and budgets. Projects can vary in scope and duration, targeting different beneficiary groups. Effective project management is essential for ensuring that resources are used efficiently and that the intended impact is achieved.
* **Attributes**:
* **ProjectID** (Primary Key): A unique identifier for each project.
* **Title**: The title of the project.
* **Description**: A detailed description of the project's objectives and activities.
* **Duration:**
* **StartDate:** The date the project begins.
* **EndDate:** The date the project is expected to conclude.
* **Budget**: The total budget allocated for the project.
* **Status**: Current status of the project (e.g., planning, ongoing, completed).
* **BeneficiaryCount**: The number of beneficiaries served by the project.

1. **Beneficiary**

****

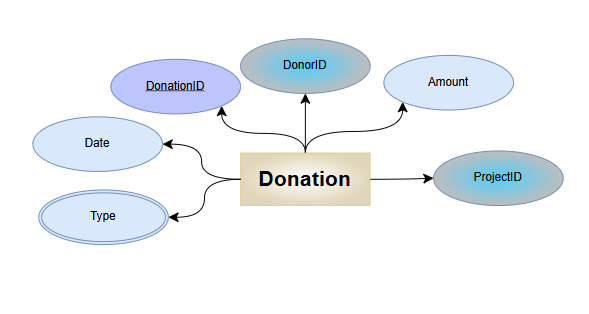
* **Description**: Beneficiaries are individuals or groups who receive assistance or services from the NGO. They may be vulnerable populations, such as low-income families, children, or marginalized communities. Understanding the demographics and needs of beneficiaries is crucial for tailoring services and measuring impact.
* **Attributes**:
* **BeneficiaryID** (Primary Key): A unique identifier for each beneficiary.
* **Name**:  Composite attribute that consists of FirstName, MiddleName (if applicable), and LastName, allowing for a complete representation of the beneficiary's name.
* **ContactInfo**: Composite attribute that includes Email and PhoneNumbers, providing multiple ways to contact the beneficiary.
* **Demographics**: Multi-valued attribute that includes multiple pieces of information such as Age, Gender, and Location.
* **ServicesReceived**: Multi-valued attribute that includes a list of services or assistance provided to the beneficiary.

1. **Staff**

****

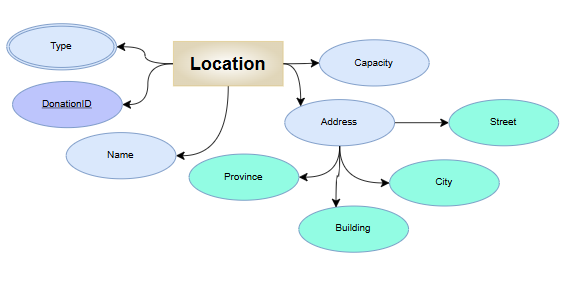
* **Description**: Staff members are employees of the NGO responsible for managing operations, projects, and administrative tasks. They play a vital role in ensuring that the NGO runs smoothly and that projects are executed effectively. Staff members may have specific roles, such as project managers, coordinators, or administrative personnel.
* **Attributes**:
* **StaffID** (Primary Key): A unique identifier for each staff member.
* **Name**: Composite attribute that consists of FirstName, MiddleName (if applicable), and LastName, allowing for a complete representation of the staff member's name.
* **Position**: The job title or role of the staff member within the NGO.
* **ContactInfo**: Composite attribute that includes Email and PhoneNumber, providing multiple ways to contact the staff member.
* **ProjectAssignments**: Multi-valued attribute that includes a list of projects the staff member is currently managing or involved in.

1. **Donation**

****

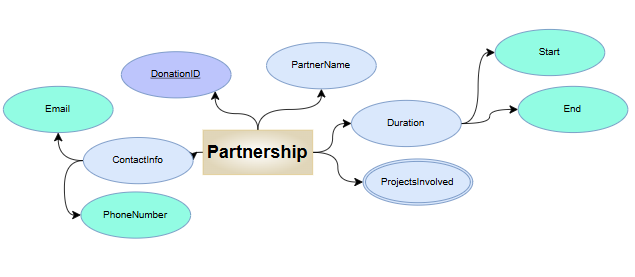
* **Description**: Donations are records of financial or in-kind contributions made by donors to support the NGO's activities. Tracking donations is essential for financial management and reporting. Each donation can be linked to specific projects, allowing the NGO to demonstrate the impact of contributions.
* **Attributes**:
* **DonationID** (Primary Key): A unique identifier for each donation.
* **DonorID** (Foreign Key): A reference to the donor who made the contribution.
* **Amount**: The monetary value of the donation.
* **Date**: The date when the donation was made.
* **Type**: Multi-valued attribute that can include multiple types such as Monetary and In-Kind, making it a multi-valued attribute.
* **ProjectID** (Foreign Key): A reference to the project associated with the donation.

1. **Location**

****

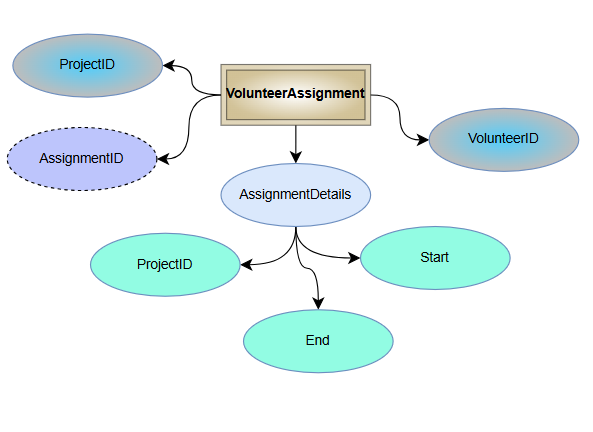
* **Description**: Locations refer to the physical sites where the NGO operates, including its main office and various field sites. Understanding the locations is essential for project planning and resource allocation. Locations can vary in capacity and purpose, serving as offices, community centers, or project implementation sites.
* **Attributes:**
* **LocationID**(Primary Key): A unique identifier for each location.
* **Name**: The name of the location (e.g., Main Office, Community Center).
* **Address**: Composite attribute that includes multiple components: Street, City, Province, and Building, providing detailed location information.
* **Type**: Multi-valued attribute that include multiple types of location such as Office, Field Site, and Community Center.
* **Capacity**: The maximum number of people the location can accommodate.

1. **Partnership**

****

* **Description**: Partnerships refer to collaborations with other organizations or entities to achieve common goals. These partnerships can enhance the NGO's capacity to deliver services, share resources, and reach a broader audience. Effective partnerships can lead to increased funding opportunities and improved project outcomes.
* **Attributes**:
* **PartnershipID** (Primary Key): A unique identifier for each partnership.
* **PartnerName**: The name of the partner organization.
* **ContactInfo**: Composite attribute that includes Email and PhoneNumber of the partner organization, providing multiple ways to contact the partner.
* **Duration:**
* **StartDate:** The date when the partnership begins.
* **EndDate:** The date when the partnership is expected to conclude.
* **ProjectsInvolved**: Multi-valued attribute that includes a list of projects that are part of the partnership, indicating the collaborative efforts.

1. **Volunteer\_Assignment**(Weak entity)

****

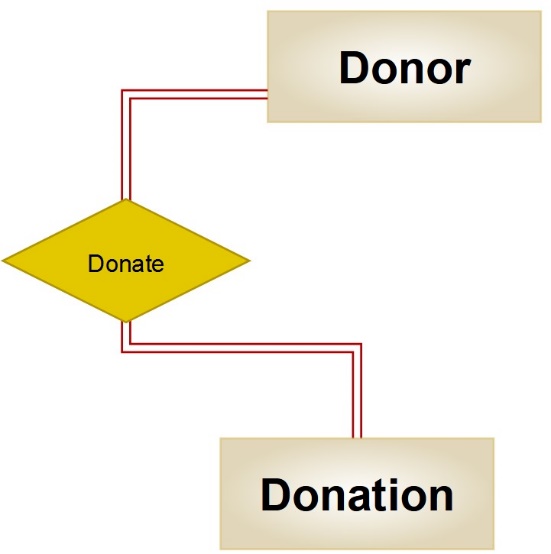
* **Description**: The Volunteer\_Assignment entity records the assignments of volunteers to various projects within the NGO. This entity is vital for managing volunteer engagement, tracking their contributions, and ensuring that projects are adequately staffed.
* **Attributes**:
* **Assignment\_ID**: A unique identifier for each volunteer assignment, allowing for precise tracking of volunteer involvement.
* **Project\_ID**: A foreign key referencing the project to which the volunteer is assigned, linking volunteers to specific initiatives.
* **Volunteer\_ID**: A foreign key referencing the volunteer assigned to the project, ensuring that the NGO can track individual contributions and manage volunteer records.
* **AssignmentDetails:** Composite attribute that includes multiple components:
* **StartDate:** The date the assignment begins.
* **EndDate:** The date the assignment ends.
* **ProjectID:** The ID of the project the volunteer is assigned to, linking it to the Project entity.

The **Volunteer\_Assignment** entity is classified as a weak entity because it relies on the **Volunteer** entity for its identification. Specifically, the **Assignment\_ID** is derived from the combination of **Volunteer\_ID** and **Project\_ID**, which together form a composite primary key that uniquely identifies each assignment. This dependency highlights that the **Volunteer\_Assignment** entity cannot exist independently; it must be associated with a specific volunteer and project. As such, it serves to track the assignments of volunteers to various projects within the NGO, ensuring effective management of volunteer engagement and contributions.

**Relationships Between Entities**

The relationships between these entities will be crucial for the functionality of the application. Here are some entity relationships:

1. **Donor to Donation:**

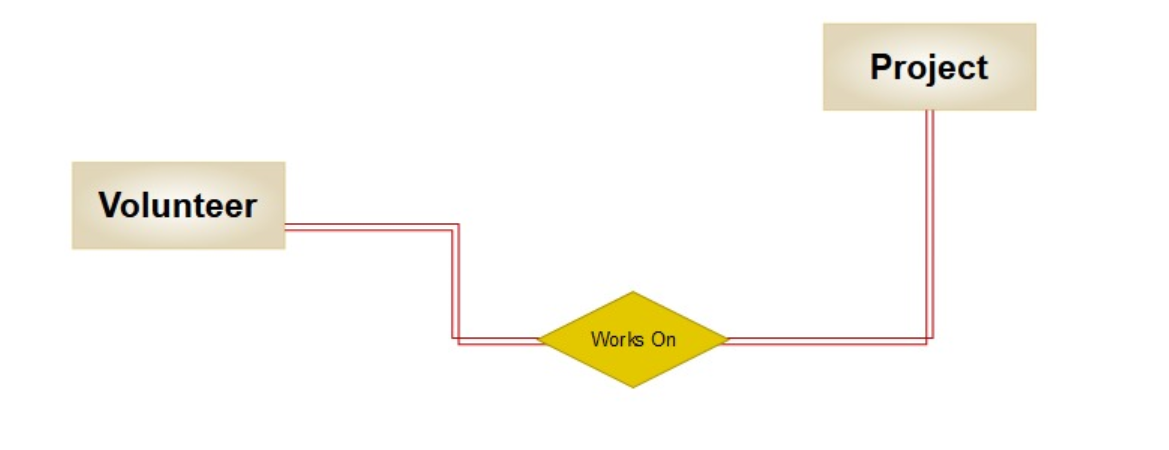
****

A donor can make multiple donations, but each donation is linked to a single donor. This relationship is one-to-many (1:N). For instance, a corporate donor may contribute funds multiple times throughout the year to support UFL's initiatives, such as providing food and medical care to displaced families. This allows UFL to track the total contributions from each donor over time, facilitating better relationship management and reporting on the impact of their support.

1. **Donation to Project:**

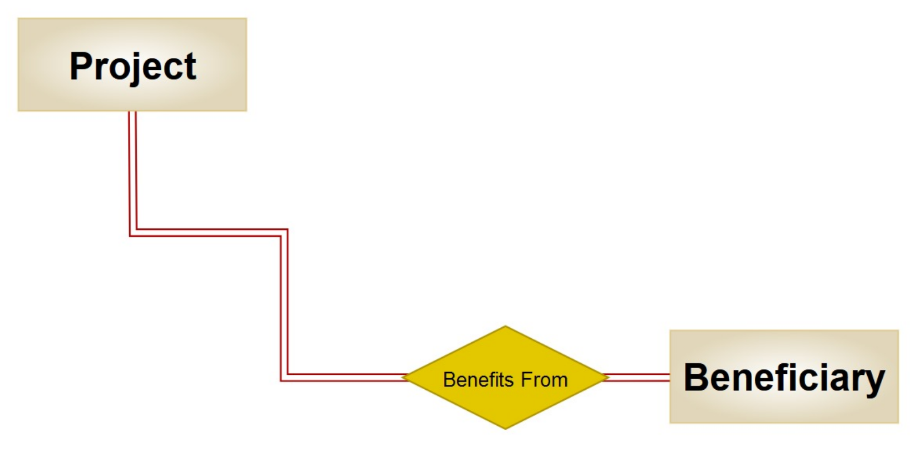
A donation can be associated with a specific project, but a project can receive multiple donations from different donors. This relationship is many-to-one (N:1). For example, a project aimed at providing shelter for families affected by the war may receive donations from various donors, each contributing different amounts. This structure enables UFL to allocate funds effectively and demonstrate the impact of each donation on specific projects, such as rebuilding homes or providing essential supplies.

1. **Volunteer to Project:**

****

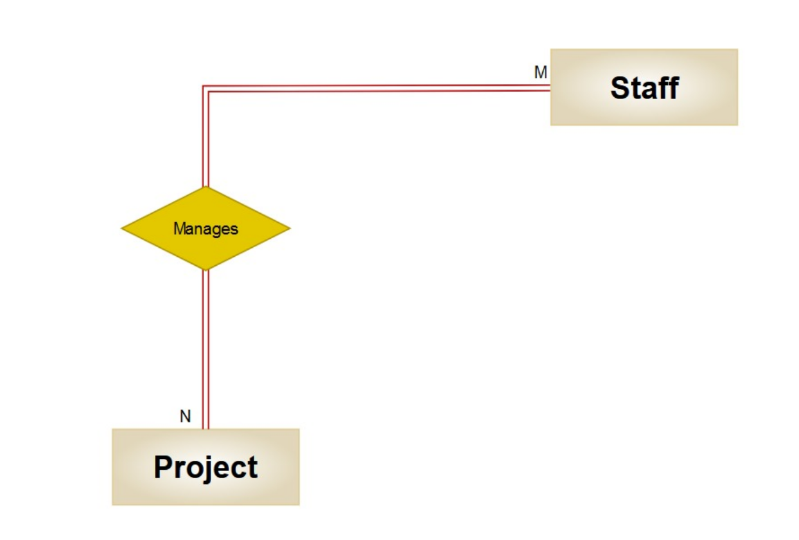
A volunteer can participate in multiple projects, and each project can have multiple volunteers. This relationship is many-to-many (M:N). For instance, a volunteer may assist in both a food distribution project and a psychological support initiative for trauma-affected individuals. This flexibility allows UFL to utilize volunteers' diverse skills across various initiatives, maximizing their impact and engagement in the community.

1. **Beneficiary to Project:**

****

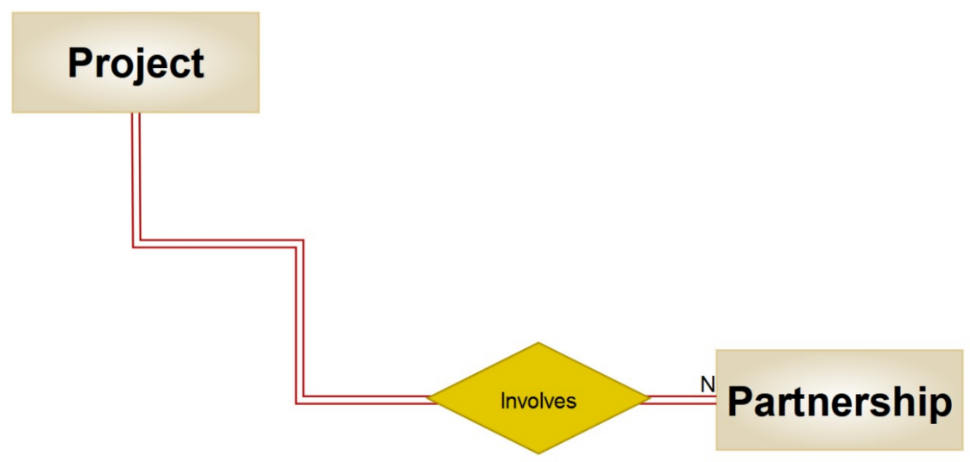
A beneficiary can receive services from multiple projects, and each project can serve multiple beneficiaries. This relationship is many-to-many (M:N). For example, a beneficiary may receive assistance from both a housing project and a medical care program. This relationship highlights the interconnectedness of services provided by UFL and ensures that beneficiaries can access comprehensive support tailored to their needs, especially in the aftermath of conflict.

1. **Staff to Project:**

****

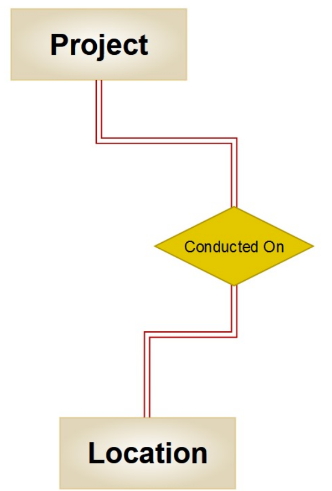
A staff member can manage multiple projects, and each project can have multiple staff members involved. This relationship is many-to-many (M:N). For instance, a project manager may oversee several initiatives aimed at providing food, shelter, and medical care, while different staff members may be assigned to various roles within those projects. This structure allows for efficient resource allocation and ensures that projects benefit from diverse expertise and oversight.

1. **Partnership to Project:**

****

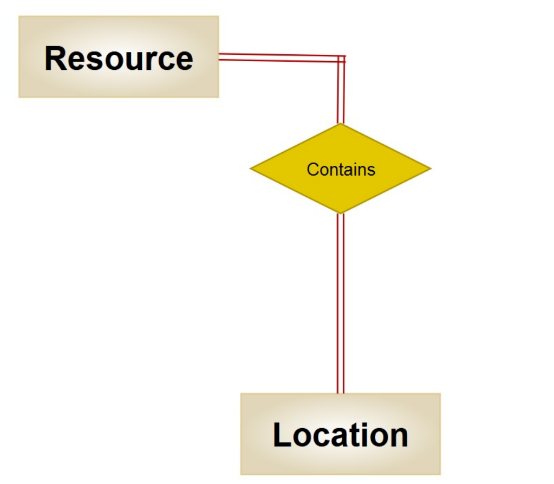
A partnership can involve multiple projects, and a project can have multiple partnerships. This relationship is many-to-many (M:N). For instance, UFL may partner with local businesses to provide resources for a specific project while also collaborating with international NGOs on different initiatives aimed at community rebuilding. This collaborative approach can lead to shared resources, expertise, and increased funding opportunities, ultimately enhancing project outcomes.

1. **Location to Project:**

****

A location can host multiple projects, and each project can be conducted at one or more locations. This relationship is many-to-many (M:N). For example, a community health initiative may take place at various community centers across Lebanon, while a housing project may be implemented in different neighborhoods affected by the war. This relationship allows UFL to effectively plan and allocate resources based on geographical needs and project requirements.

1. **Location to Resource:**

****

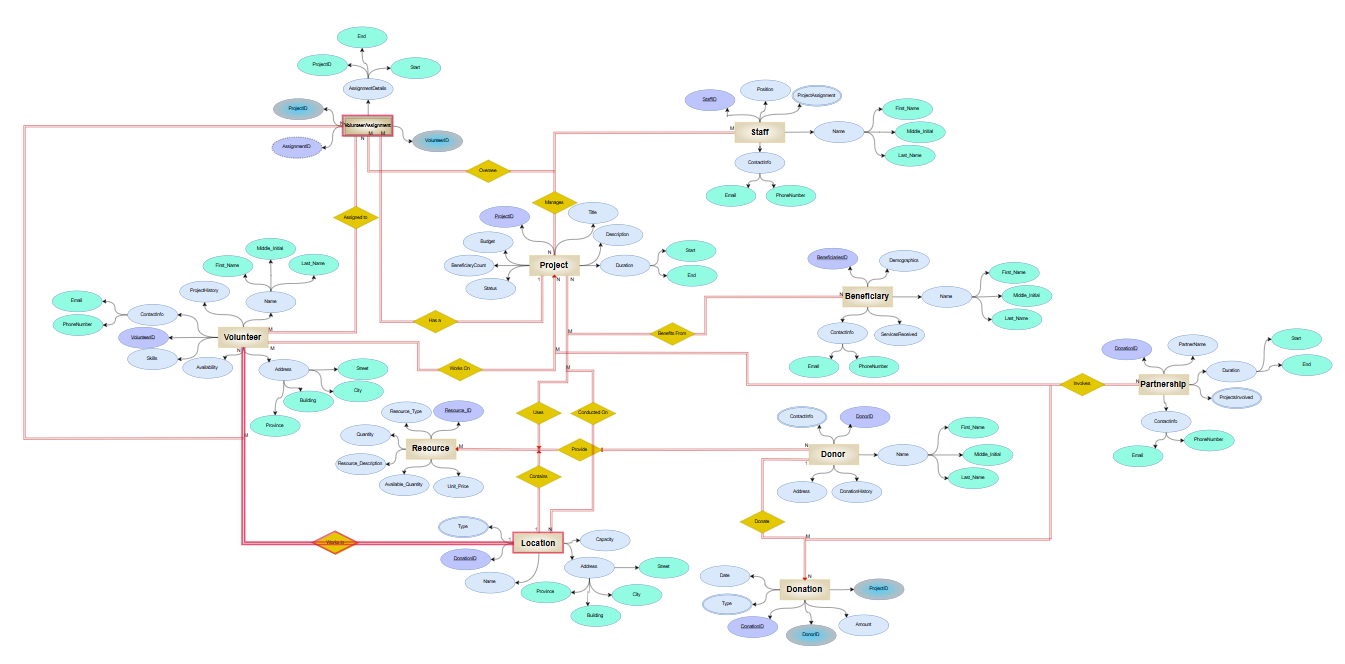
A location can store multiple resources, and each resource can be associated with one or more locations. This relationship is many-to-many (M:N). For instance, a resource like educational materials may be stored at various community centers, while a specific location may house multiple types of resources, such as food supplies and medical equipment. This relationship ensures efficient resource management and accessibility across different operational sites, which is vital for UFL's mission to support the community.

**Potential Challenges**

While implementing the NGO database application, several challenges may arise:

1. **User Adoption**:
   * Resistance to change from staff and volunteers accustomed to existing processes may hinder the adoption of the new system.
   * To mitigate this, comprehensive training and support should be provided to ensure users feel comfortable with the new application.
2. **Data Migration**:
   * Migrating existing data from legacy systems or spreadsheets into the new database can be complex and time-consuming.
   * A well-planned data migration strategy should be developed, including data cleaning and validation processes.
3. **Budget Constraints**:
   * Limited financial resources may impact the scope of the project, including software, hardware, and training costs.
   * Seeking grants or partnerships with tech companies may help alleviate some financial burdens.
4. **Technical Issues**:
   * Technical challenges, such as software bugs or server downtime, can disrupt operations.
   * Having a dedicated IT support team and a contingency plan in place will help address these issues promptly.
5. **Compliance and Regulations**:
   * NGOs must comply with data protection regulations (e.g., GDPR, HIPAA) when handling personal information.
   * Ensuring that the database application adheres to these regulations will be crucial to avoid legal issues.

**ER diagram for "Unity for Lebanon"(UFL) NGO:**

****

**ER to Relational Mapping Algorithms**

**Step 1: Mapping of Regular Entity Types**

**Database Design**

The database will be structured to reflect the above relationships:

**Tables and Attributes**

In the first step, each regular entity type is mapped into a relational schema. Every entity will have its own relation, including all of its simple attributes and a single primary key, which is underlined. The following regular entities are designed for the UFL NGO database: Resource, Donor, Volunteer, Project, Beneficiary, Staff, Donation, Location, and Partnership.

**Resource Table**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ResourceID | Resource\_Type | Resource\_Description | Available\_Quantity | Unit\_Price | Quantity |

The Resource entity contains simple attributes essential for tracking the resources used by UFL. Each resource has a unique identifier, Resource\_ID, which is the primary key. Other attributes include Resource\_Type (e.g., Human, Financial, Material), Resource\_Description (providing additional context), Quantity (total amount available), Available\_Quantity (current quantity in stock), and Unit\_Price (for budgeting purposes). This structure ensures efficient resource management across UFL’s various projects and initiatives.

**Donor Table**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| DonorID | First\_Name | Middle\_Name | Last\_Name | Address |
| ContactInfo | DonationHistory |

The Donor entity records information about individuals or organizations that contribute to UFL’s projects. Each donor is identified by DonorID, the primary key. Name is a composite attribute composed of FirstName, MiddleName, and LastName. ContactInfo is a multi-valued attribute to capture multiple contact methods. Address is also composite, including details like street, city, and province. DonationHistory records past donations, helping UFL maintain strong relationships with supporters.

**Volunteer Table**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| VolunteerID | First\_Name | Middle\_Name | Last\_Name | Email | Phone\_Number | Skills |
| Availability | Province | Building | City | Street | ProjectHistory |

The Volunteer entity is essential for managing UFL’s volunteer network. VolunteerID serves as the primary key. Name is composite, including FirstName, MiddleName, and LastName. ContactInfo is a composite attribute to store email and phone numbers. Skills is a multi-valued attribute, listing the volunteer’s abilities. Availability records their working hours. ProjectHistory tracks past projects the volunteer has participated in, and Address is composite, including province, building, city, and street detailing the exact location.

**Project Table**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ProjectID | Title | Description | StartDate | EndDate | Budget | Status | BeneficiaryCount |

The Project entity details UFL’s initiatives, each identified by ProjectID (primary key). The Title and Description provide context about the project’s objectives. StartDate and EndDate mark the timeline, while Budget indicates the financial allocation. Status reflects the project’s progress (e.g., planning, ongoing, completed), and BeneficiaryCount shows the number of people the project serves.

**Beneficiary Table**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| BeneficiaryID | First\_Name | Middle\_Name | Last\_Name | Email | Phone\_Number |
| Demographics | ServicesReceived |

The Beneficiary entity represents individuals who receive assistance from UFL. BeneficiaryID is the primary key, and Name is a composite attribute including FirstName, MiddleName, and LastName. ContactInfo allows for multiple ways to reach beneficiaries, and Demographics includes details like email and phone number. ServicesReceived is a multi-valued attribute, listing all services or support provided to the beneficiary.

**Staff Table**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| StaffID | First\_Name | Middle\_Name | Last\_Name | Position | Email | Phone\_Number |
| ProjectAssignments |

The Staff entity manages UFL’s employees, each uniquely identified by StaffID. Name is composite, including FirstName, MiddleName, and LastName; and Position specifies the staff member’s role (e.g., coordinator, manager). ContactInfo is composite for email and phone numbers. ProjectAssignments is a multi-valued attribute, listing projects the staff member is involved in.

**Donation Table**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| DonationID | DonorID (FK) | Amount | Date | Type | ProjectID (FK) |

The Donation entity records contributions from donors, with DonationID as the primary key. DonorID is a foreign key linking to the Donor entity. Amount and Date track the contribution details, while Type (a multi-valued attribute) indicates if it’s monetary or in-kind. ProjectID links the donation to specific UFL projects, highlighting its impact(Foreign key).

**Location**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| LocationID | Name | Province | Building | City |
| Street | Type | Capacity |

The Location entity tracks UFL’s physical sites, each identified by LocationID. Name specifies the location type (e.g., Main Office, Field Site). Address is composite, including province, building, city, and street detailing the exact location. Type is multi-valued, listing various uses (e.g., Office, Community Center), and Capacity indicates the maximum occupancy.

**Partnership Table**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| PartnershipID | PartnerName | Email | Phone\_Number | StartDate | EndDate | ProjectsInvolved |

The Partnership entity documents UFL’s collaborations with other organizations. PartnershipID is the primary key. PartnerName identifies the partner organization, while ContactInfo provides their email and phone details. StartDate and EndDate define the duration of the partnership. ProjectsInvolved is multi-valued, listing all joint projects.

**Step 2: Mapping of Weak Entity Types**

Identifying weak entities, which rely on the primary key of another entity for identification, and map pingthem with foreign keys.

**Volunteer\_Assignments**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Assignment\_ID | Project\_ID | Volunteer\_ID | Start\_Date | End\_Date |

Here, Assignment\_ID is redundant since the weak entity relies on Volunteer\_ID and Project\_ID. So we have a composite primary key (Volunteer\_ID, Project\_ID).

**Step 3: Mapping of Binary 1:1 Relation Types**

For each **1:1** relationship, adding a foreign key from one entity to the other.

If the **1:1** relationship is total on both sides, we will be choosing one entity to contain the foreign key of the other.

**But there exist no 1:1 relationship in our ER model**

**Step 4: Mapping of Binary 1:N Relationship Types**

For each **1:N** relationship, we will add a foreign key in the entity on the “many” side that references the primary key of the entity on the “one” side.

1. Donor to Donation:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| DonorID (FK) | DonationID (PK) | Amount | ProjectID | Date | Type |

Each donation is made by a specific donor. By including DonorID as a foreign key in the Donation table, each donation record references the donor who made it, ensuring that the source of each donation is tracked.

2. Location to Project:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ProjectID | Title | Description | Duration | Budget | BeneficiaryCount | Status | LocationID(FK) |

Each project occurs at a specific location. By including LocationID as a foreign key in the Project table, each project record can reference the location where it takes place, ensuring proper tracking of project locations.

**Step 5: Mapping of Binary M:N Relationship Types**

For each **M:N** relationship, create a new relation to represent the relationship. The primary key of this new relation is the combination of the primary keys of the participating entities.

1. Volunteer to Project: **Volunteer\_Project**

|  |  |  |  |
| --- | --- | --- | --- |
| VolunteerID + ProjectID (PK) | AssignmentDetails | StartDate | EndDate |

This table links volunteers to projects they are assigned to. Each record in Volunteer\_Project indicates a specific volunteer’s assignment to a specific project.

2. Beneficiary to Project: **Project\_Beneficiary**

|  |  |
| --- | --- |
| BeneficiaryID + ProjectID (PK) | ServiceDetails |

This table links projects to their beneficiaries. Each record in Project\_Beneficiary shows which beneficiaries are linked to which projects

3. Location to Resource: **Location\_Resource**

|  |  |  |
| --- | --- | --- |
| LocationID + ResourceID (PK) | Quantity | Type |

This table links resources to locations where they are stored or used. Each record in Location\_Resource would specify the quantity of a resource at a specific location

4. Partnership to Project: **Project\_Partnership**

|  |  |  |  |
| --- | --- | --- | --- |
| ProjectID + PartnershipID (PK) | PartnershipDetails | StartDate | EndDate |

This table links projects to their partnerships. Each record in Project\_Partnership indicates a partnership between a specific project and a partner organization.

**Step 6: Mapping of Multivalued Attributes**

For each multivalued attribute, we will be creating a new relation containing the attribute and the primary key of the entity it belongs to.

**1.** **Donor\_ContactInfo:**

|  |  |
| --- | --- |
| DonorID | ContactInfo |

This table records the various contact information of donors, such as phone numbers and email addresses. Each record ensures that multiple contact details for a donor are properly managed and linked back to the donor entity.

**2. Volunteer\_Skills:**

|  |  |
| --- | --- |
| VolunteerID | Skill |

This table stores the skills of each volunteer. Each record represents a specific skill possessed by a volunteer, ensuring that multiple skills for a volunteer are efficiently tracked and linked back to the volunteer entity.

**3. Volunteer\_Project\_History:**

|  |  |
| --- | --- |
| VolunteerID | ProjectID |

This table logs the history of projects that a volunteer has been involved in. Each record links a volunteer to a project, allowing for a detailed overview of the volunteer’s participation over time.

**Step 7: Mapping of N-ary Relationship Types**

For any N-ary relationships (relationships involving more than two entities), we will be creating a new relation with the primary keys of all entities involved as foreign keys.

In this project, if an N-ary relationship exists (such as a collaboration between a donor, project, and staff), we will define it similarly by including foreign keys referencing each entity.

**Not applicable in this project**

**FINAL STEP: Final Displays**

**Resource Table**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ResourceID** | **Resource\_Type** | **Resource\_Description** | **Available\_Quantity** | **Unit\_Price** |

**Donor Table**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **DonorID** | **First\_Name** | **Middle\_Name** | **Last\_Name** | **ContactInfo** |
| **Address** | **DonationHistory** |

**Volunteer Table**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **VolunteerID** | **First\_Name** | **Middle\_Name** | **Last\_Name** | **Email** | **Phone\_Number** | **Skills** |
| **Availability** | **Province** | **Building** | **City** | **Street** | **ProjectHistory** |

**Project Table**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ProjectID** | **Title** | **Description** | **StartDate** | **EndDate** | **Budget** | **Status** | **BeneficiaryCount** |

**Beneficiary Table**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **BeneficiaryID** | **First\_Name** | **Middle\_Name** | **Last\_Name** | **Email** | **Phone\_Number** |
| **Demographics** | **ServicesReceived** |

**Staff Table**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **StaffID** | **First\_Name** | **Middle\_Name** | **Last\_Name** | **Position** | **Email** |
| **Phone\_Number** | **ProjectAssignments** |

**Donation Table**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **DonationID** | **DonorID (FK)** | **Amount** | **Date** | **Type** | **ProjectID(FK)** |

**Location**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **LocationID** | **Name** | **Province** | **Building** | **City** |
| **Street** | **Type** | **Capacity** |

**Partnership Table**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **PartnershipID** | **PartnerName** | **Email** | **Phone\_Number** | **StartDate** | **EndDate** |
| **ProjectsInvolved** |

**Volunteer\_Assignments**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Assignment\_ID** | **Project\_ID** | **Volunteer\_ID** | **Start\_Date** | **End\_Date** |

**Donor to Donation:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **DonorID (FK)** | **DonationID (PK)** | **Amount** | **ProjectID** | **Date** | **Type** |

**Location to Project:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ProjectID** | **Title** | **Description** | **Duration** | **Budget** | **BeneficiaryCount** | **Status** | **LocationID(FK)** |

**Volunteer to Project: Volunteer\_Project**

|  |  |  |  |
| --- | --- | --- | --- |
| **VolunteerID + ProjectID (PK)** | **AssignmentDetails** | **StartDate** | **EndDate** |

**Beneficiary to Project: Project\_Beneficiary**

|  |  |
| --- | --- |
| **BeneficiaryID + ProjectID (PK)** | **ServiceDetails** |

**Location to Resource: Location\_Resource**

|  |  |  |
| --- | --- | --- |
| **LocationID + ResourceID (PK)** | **Quantity** | **Type** |

**Partnership to Project: Project\_Partnership**

|  |  |  |  |
| --- | --- | --- | --- |
| **ProjectID + PartnershipID (PK)** | **PartnershipDetails** | **StartDate** | **EndDate** |

**Donor\_ContactInfo:**

|  |  |
| --- | --- |
| **DonorID** | **ContactInfo** |

**Volunteer\_Skills:**

|  |  |
| --- | --- |
| **VolunteerID** | **Skill** |

**Volunteer\_Project\_History:**

|  |  |
| --- | --- |
| **VolunteerID** | **ProjectID** |

**Table Structure for Promise Hospital:**

After designing the ER diagram for “Unity of Lebanon”(UFL) NGO and mapping this diagram into relational database design, now it is time to start creating the actual tables for our database on the Oracle Database Server. We will start by creating all tables and then inserting data into these tables. Finally, we will execute some queries to display the importance of the database and especially in an NGO.

**1. Build the Database**

**1.1 Tables Creation**

Resource Table:

A close-up of a code

Description automatically generated

Donor Table:

A computer code with text

Description automatically generated with medium confidence

Volunteer Table:

A close up of text

Description automatically generated

Project Table:

A white background with black text

Description automatically generated

Beneficiary Table:

A white background with black text

Description automatically generated

Donation Table:

A computer code with text

Description automatically generated with medium confidence

Partnership Table:

A close up of text

Description automatically generated

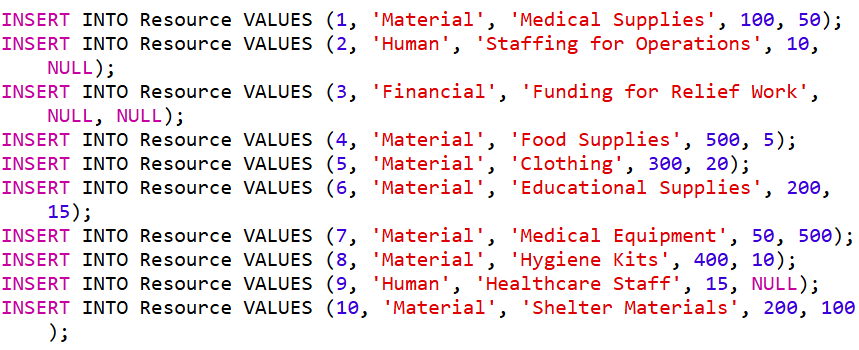
Volunteer\_Assignment Table (for M:N relationship):

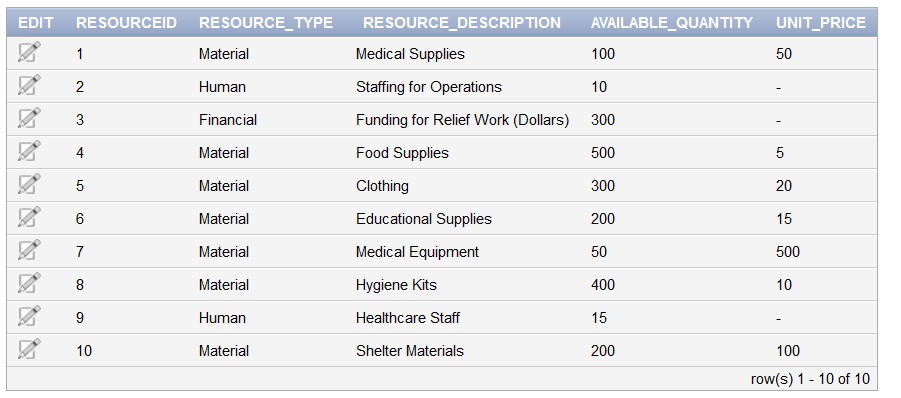
A close-up of a computer screen

Description automatically generated

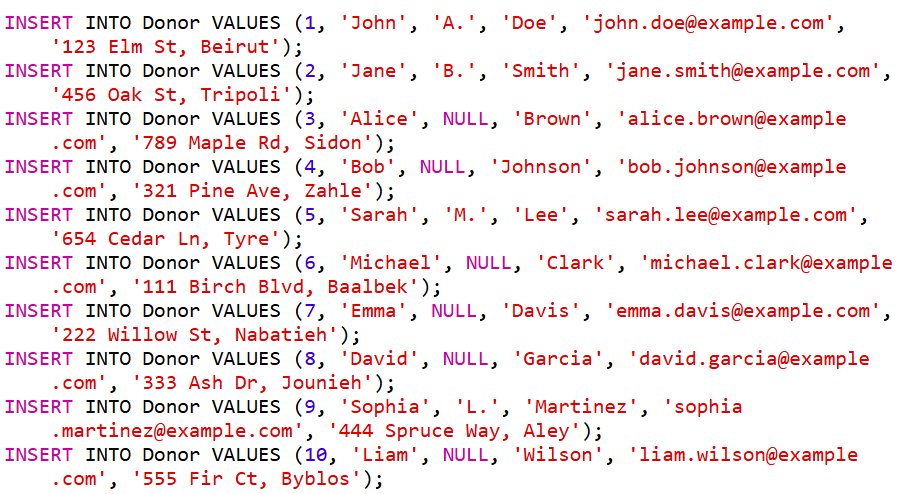
**Data Insertion:**

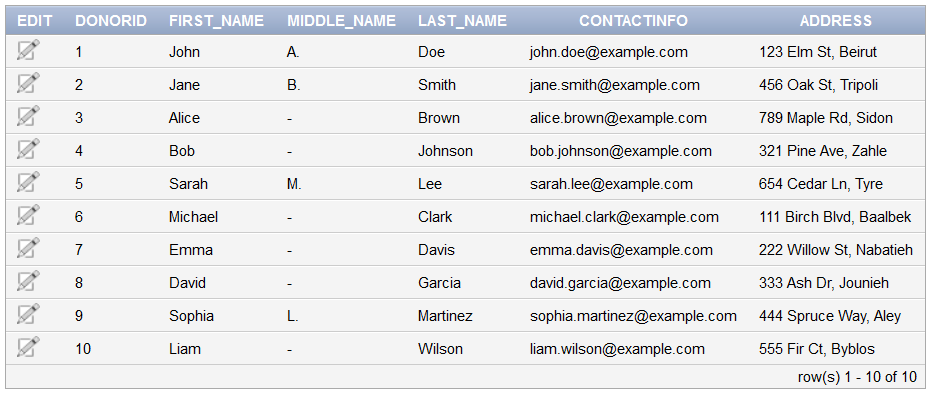
**1) Data Insertion for Resource Table:**

****

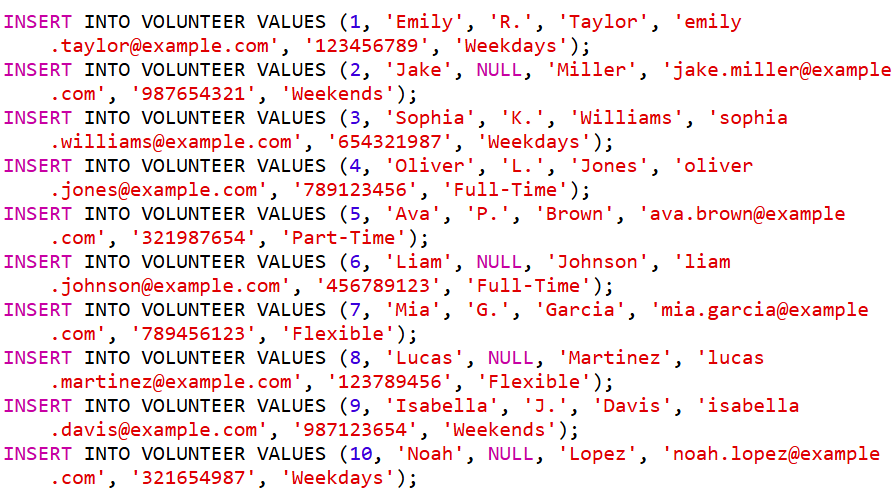


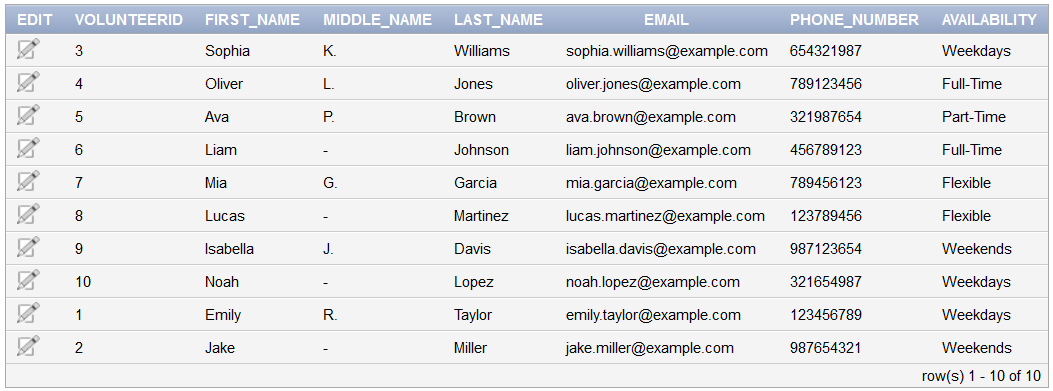
**2) Data Insertion for Donor Table:**

****

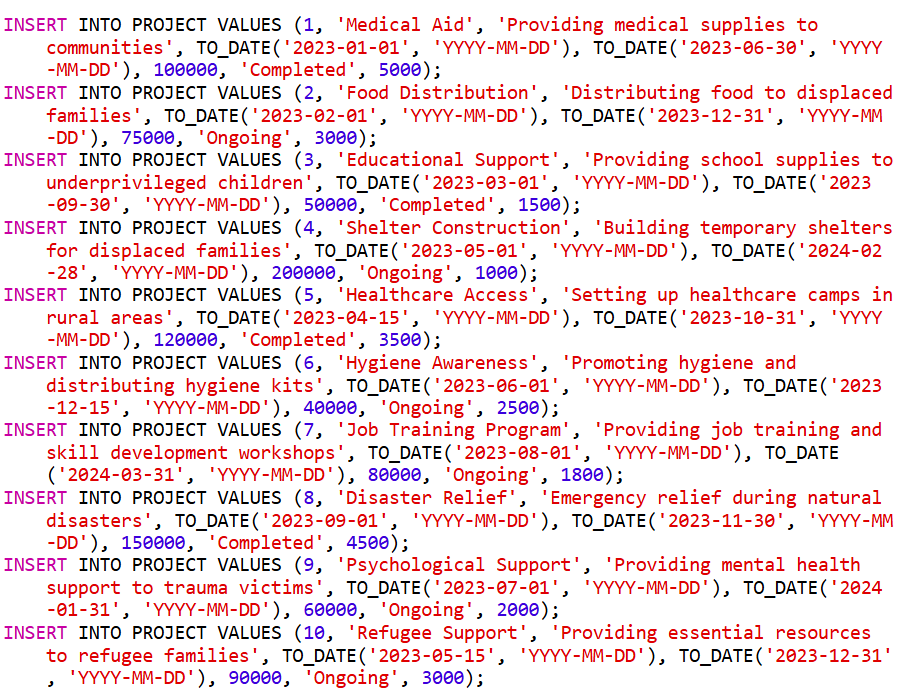


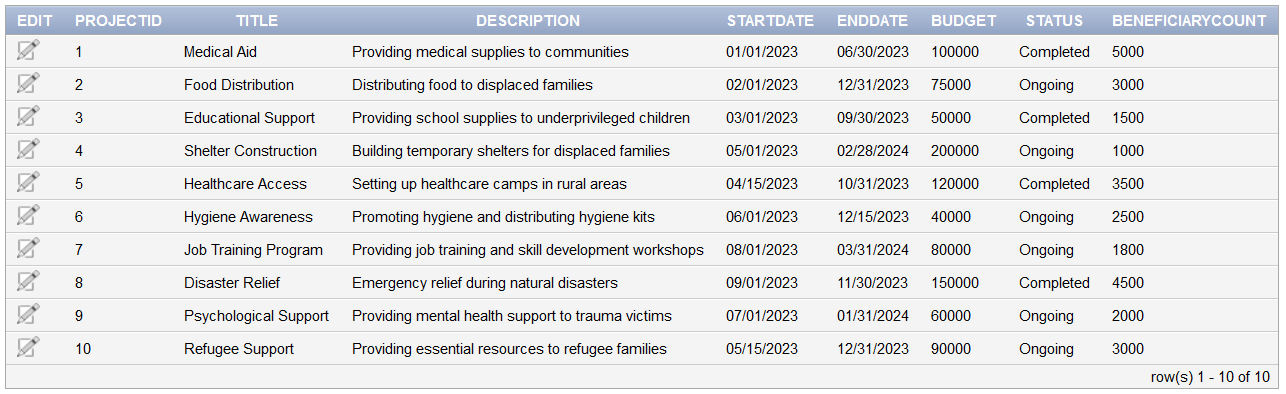
**3) Data Insertion for Volunteer Table:**

****

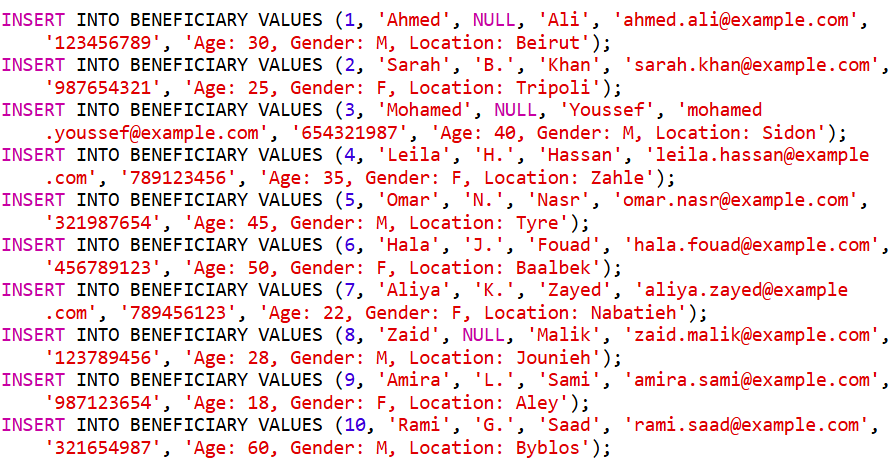


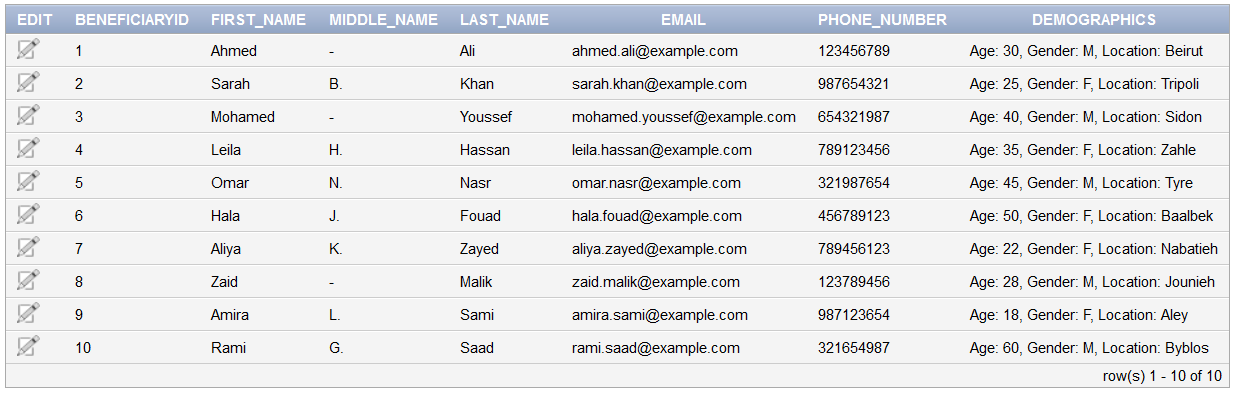
**4) Data Insertion for Project Table:**

****

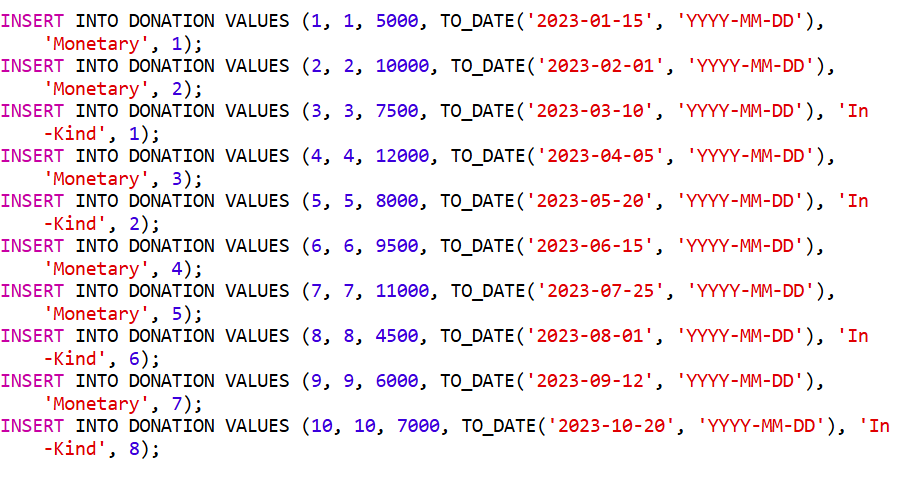


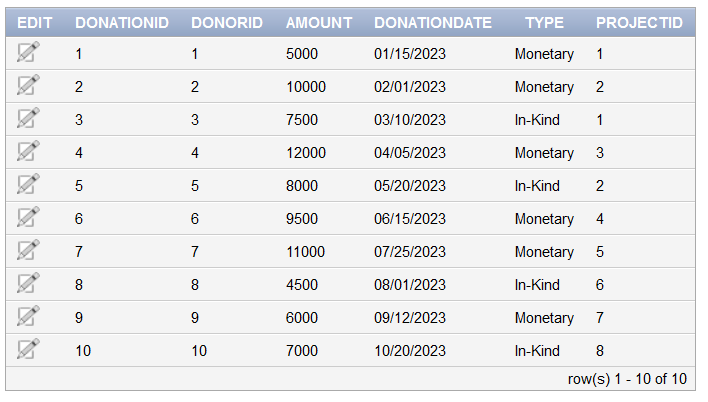
**5) Data Insertion for Beneficiary Table:**

****

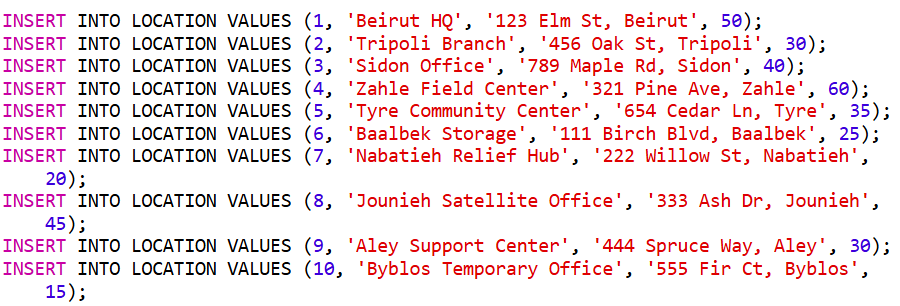


**6) Data Insertion for Donation Table:**

****



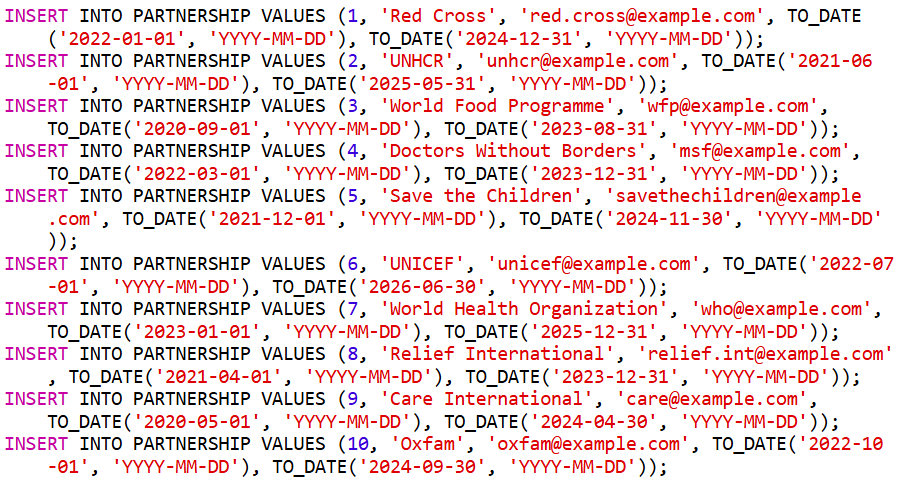
**7) Data Insertion for Location Table:**

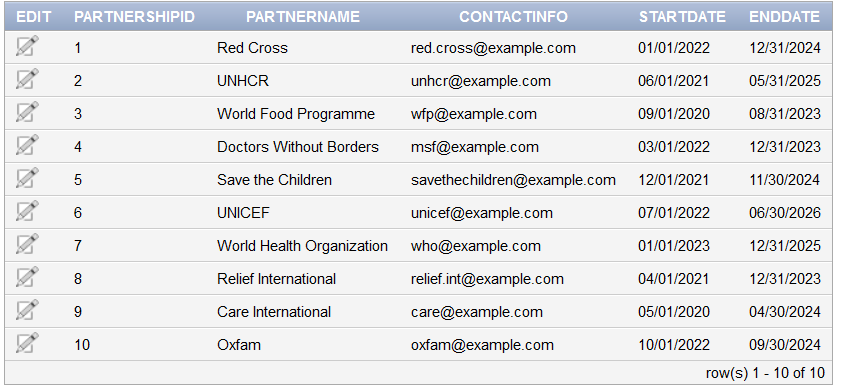
****

A screenshot of a computer

Description automatically generated

**8) Data Insertion for Partnership Table:**

****



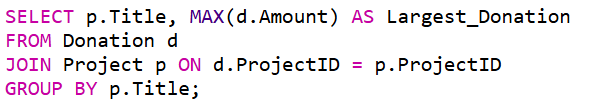
**Table Descriptions:**

After creating all the tables on the oracle database server we can view the description of each table in order to make sure everything is fine and no mistakes were made during creation of table.  
In our database we have the following tables created on the oracle database server:

**Query 1: List projects receiving the largest donation**

Purpose: Identify the project(s) that received the single largest donation amount.

SQL Query:

A screenshot of a computer

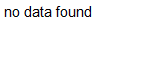
Description automatically generated

**Query 2: Retrieve donors who contributed more than once**

Purpose: Identify donors who have made more than one donation.

SQL Query:

A close up of a computer screen

Description automatically generated

**Query 3: Count projects hosted at each location**

Purpose: Display the number of projects hosted by each location.

SQL Query:

A close-up of a computer screen

Description automatically generated

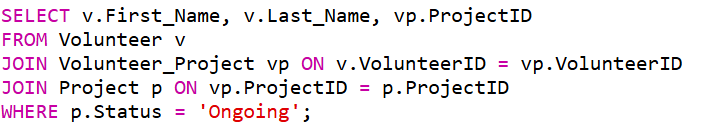
A screenshot of a computer

Description automatically generated

**Query 4: Volunteers working on ongoing projects**

Purpose: List volunteers participating in projects with the status "Ongoing".

SQL Query:

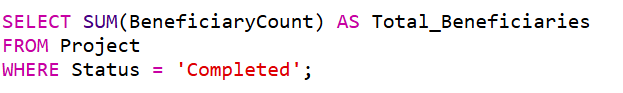


**no data found**

**Query 5: Total beneficiaries served by completed projects**

Purpose: Calculate the total number of beneficiaries served by completed projects.

SQL Query:



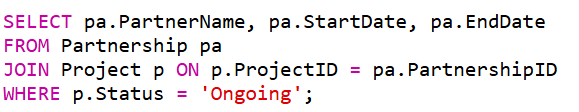
A close-up of a sign

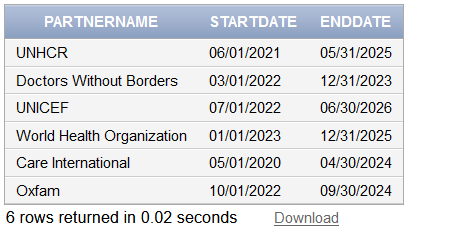
Description automatically generated

**Query 6: Partnership duration for ongoing projects**

Purpose: Retrieve the duration of partnerships linked to ongoing projects.

SQL Query:





**Query 7: Identify total value of each resource**

Purpose: List the total value of each resource that has a price.

SQL Query:

A white background with black and pink text

Description automatically generated

A screenshot of a computer

Description automatically generated

**Query 8: Donor contributions by location**

Purpose: For every location, show the total contributions received.

SQL Query:

A black text with pink and purple text

Description automatically generated

A screenshot of a computer

Description automatically generated

**Query 9: Average budget of projects by status**

Purpose: Calculate the average budget of projects for each status (e.g., ongoing, completed).

SQL Query:

A close up of words

Description automatically generated

A screenshot of a computer

Description automatically generated

**Query 10: Most active volunteers**

Purpose: Find volunteers involved in the highest number of projects.

SQL Query:

A screenshot of a computer code

Description automatically generated

A white background with black text

Description automatically generated

Volunteer\_Skills:

A screenshot of a computer

Description automatically generated

Volunteer\_Project:

A screenshot of a computer

Description automatically generated

Project\_Beneficiary:

A screenshot of a computer

Description automatically generated

Partnership\_Project:

A screenshot of a data type

Description automatically generated

Donor\_ContactInfo:

A screenshot of a computer

Description automatically generated

Volunteer\_\_Project\_History:

A screenshot of a computer

Description automatically generated

**Normalizing to 1nf:**

This form does not allow multivalued attributes, composite attributes, and their combinations to exist in a relation.

1. Only attribute values permitted are single atomic values.

2. Domain of an attribute must only include atomic values and the value of an attribute in a tuple must be a single value from the domain of that attribute.

3. Disallows having a set of values as an attribute value for a single tuple.

**Normalizing to 2nf:**

The Second normal form is based on the concept of full functional dependency. Before explaining the second form let us define some concepts used in this form and other forms also.

Functional Dependencies: A constraint between two sets of attributes from the database. The values of the Y component of a tuple in relation R depend on, or are determined by the values of an X component. We say that Y is functionally dependent on X.

Prime attribute: An attribute that is a member of a candidate key in a relation R. An attribute is called non prime if it is not a prime attribute that is, if it is not a member of any candidate key.

Full functional dependency: A functional dependency X →Y is a full functional dependency if removal of any attribute A from X means that the dependency does not hold anymore.

Partial Dependency: A functional dependency X →Y is a partial functional dependency if removal of any attribute A from X means that the dependency still holds.

A relation schema R is in the second normal form if every nonprime attribute in R is fully functionally dependent on the every key of R and every nonprime attribute A in R is not partially dependent on any key in R.

**Normalizing to 3nf:**

The third normal form is based on the concept of transitive dependency. So let us first define a transitive dependency.

Transitive Dependency: A functional dependency X →Y in a relation schema R is a transitive dependency if there exists a set of attributes Z in R that is neither a candidate key nor a subset of any key of R, and both X →Z and Z →Y hold.

A relation schema R is in the third normal form if it satisfies the second normal form and no nonprime attribute of R is transitively dependent on the primary key. For every nontrivial functional dependency X→Y either X should be a super key or Y is a prime attribute

**Normalizing to BCNF:**

The Boycee Codd normal form is a stricter form than the third normal form. The BCNF differs from the definition of the third normal form in only one condition. The third normal form allows the right hand side of the functional dependency to be a prime attribute while BCNF does not allow that.

**Resource Table**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ResourceID** | **Resource\_Type** | **Resource\_Description** | **Available\_Quantity** | **Unit\_Price** |

**Attributes are directly related to the resources (ResourceID, Resource\_Type, Resource\_Description, Available\_Quantity, Unit\_Price). This avoids redundancy by ensuring that each resource is uniquely identified by ResourceID, adhering to BCNF.**

**Donor Table**

|  |  |  |  |
| --- | --- | --- | --- |
| **DonorID** | **First\_Name** | **Middle\_Name** | **Last\_Name** |

**Contains personal information about donors (DonorID, First\_Name, Middle\_Name, Last\_Name). Normalization separates address and contact details into separate tables, reducing redundancy.**

**Donor Main Address Table**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **DonorID** | **AddressType** | **Street** | **City** | **Province** | **PostalCode** |

**Each address is stored separately from the donor, identified by DonorID and AddressType. This allows a donor to have multiple addresses without duplication.**

**Donation Primary Contact Details Table**

|  |  |  |
| --- | --- | --- |
| **DonorID** | **ContactType** | **ContactValue** |

**Contact information is split by ContactType and ContactValue, ensuring one-to-many relationships for contact types (e.g., phone, email).**

**Donation History Table**

|  |  |  |
| --- | --- | --- |
| **DonorID** | **DonationID** | **DonationDate** |

**Tracks donor contributions (DonorID, DonationID, DonationDate) while avoiding duplication of donor and donation information across records.**

**Volunteer Table**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **VolunteerID** | **First\_Name** | **Middle\_Name** | **Last\_Name** | **Email** | **Phone\_Number** |

**Captures volunteer information (VolunteerID, personal details) without duplicating availability, skills, or assignments.**

**Volunteer Main Address Table**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **VolunteerID** | **Building** | **City** | **Street** | **Province** |

**Separates address details by VolunteerID, allowing each volunteer to have multiple addresses.**

**Volunteer Availability Table**

|  |  |  |  |
| --- | --- | --- | --- |
| **VolunteerAvailabilityID** | **VolunteerID** | **AvailableDate** | **AvailableTime** |

**Tracks availability independently of the volunteer's primary details, ensuring no redundancy.**

**Volunteer Skill Table**

|  |  |  |
| --- | --- | --- |
| **VolunteerSkillID** | **VolunteerID** | **Skill** |

**Links VolunteerID to their skills, ensuring that skills are not stored redundantly within the main volunteer table.**

**Project Table**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ProjectID** | **Title** | **Description** | **StartDate** | **EndDate** | **Budget** | **Status** | **BeneficiaryCount** |

**Defines project attributes (ProjectID, Title, Description, etc.). Beneficiaries, volunteers, and staff relationships are normalized into separate tables.**

**Beneficiary Table**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **BeneficiaryID** | **First\_Name** | **Middle\_Name** | **Last\_Name** | **Community** |

**Separates personal beneficiary data (BeneficiaryID, names) from demographic and contact details.**

**Beneficiary Main Phone Number**

|  |  |
| --- | --- |
| **BeneficiaryID** | **Phone\_Number** |

**Stores phone numbers separately to allow for multiple numbers per beneficiary.**

**Beneficiary Demographics Table**

|  |  |  |  |
| --- | --- | --- | --- |
| **BeneficiaryID** | **Age** | **Gender** | **Ethnicity** |

**Tracks demographic attributes (e.g., age, gender) independently of personal details, enabling easy updates without redundancy.**

**Beneficiary’s Main Email**

|  |  |
| --- | --- |
| **BeneficiaryID** | **Email** |

**Stores email addresses separately to support multiple emails per beneficiary.**

**Beneficiary Service Table**

|  |  |
| --- | --- |
| **BeneficiaryID** | **Service** |

**Links beneficiaries to services, removing any direct dependency on the BeneficiaryID from the Beneficiary table**

**Staff Table**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **StaffID** | **First\_Name** | **Middle\_Name** | **Last\_Name** | **Position** | **Email** |
| **Phone\_Number** |

**Captures staff details, normalizing their roles and project assignments into separate relational tables.**

**Staff Project Relation Table (Supervises)**

|  |  |  |
| --- | --- | --- |
| **ProjectStaffID** | **StaffID** | **ProjectID** |

**Tracks which projects are supervised by which staff members, ensuring no direct dependency between staff and projects in other tables.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **DonationID** | **DonorID** | **Amount** | **Type** | **ProjectID** |

**Donation Table**

**Breaks down donation attributes (DonationID, Amount, Type) while linking them to donors and projects in other tables.**

**Location**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **LocationID** | **IdentifiableLandmark** | **Name** | **Province** | **Building** | **City** |
| **Street1** | **Street2** | **Type** | **Capacity** |

**Stores location details (LocationID, address attributes) without associating them directly with resources or projects, enabling many-to-many relationships.**

**Partnership Table**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **PartnershipID** | **PartnerName** | **Email** | **Phone\_Number** | **StartDate** | **EndDate** |

**Captures partner organization data (PartnershipID, PartnerName, etc.), normalized to avoid redundant storage of project-specific partnership details.**

**Partnership Project Relation Table (Sponsors)**

|  |  |  |  |
| --- | --- | --- | --- |
| **ProjectPartnershipID** | **PartnershipID** | **ProjectID** | **ValueOfAssistance** |

**Tracks the relationship between partnerships and projects (ProjectPartnershipID, PartnershipID, ProjectID).**

**Volunteer\_Assignments**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Assignment\_ID** | **Project\_ID** | **Volunteer\_ID** | **Start\_Date** | **End\_Date** |

**Tracks individual volunteer assignments for projects, ensuring no redundancy in volunteer-project relationships.**

**Donor to Donation Relation Table (Donated)**

|  |  |  |
| --- | --- | --- |
| **DonorID** | **DonationID** | **Notes** |

**Manages the relationship between donors and their donations, providing notes for context without duplicating donor or donation details.**

**Donation Project Relation Table (Used on)**

|  |  |
| --- | --- |
| **DonationID** | **ProjectID** |

**Tracks how donations are used across projects, separating donation allocation from the donation or project tables.**

**Location to Project Relation Table (Located at)**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ProjectLocationID** | **ProjectID** | **LocationID** | **Description** | **Duration** | **Budget** | **BeneficiaryCount** |
| **Status** | **Title** |

**Links projects to their locations (e.g., venue), detailing budget, duration, and status.**

**Volunteer Project Relation Table (Volunteers at)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **VolunteerProjectID** | **VolunteerID** | **ProjectID** | **StartDate** | **EndDate** |

**Tracks the relationship between volunteers and projects (VolunteerProjectID, VolunteerID, ProjectID).**

**Volunteer Assignment Relation (Work needed)**

|  |  |  |
| --- | --- | --- |
| **AssignmentID** | **VolunteerID** | **AssignmentDetail** |

**Tracks specific work details for each volunteer, ensuring no duplication in assignments.**

**Beneficiary to Project: (Benefits from)**

|  |  |  |
| --- | --- | --- |
| **BenefitID** | **BeneficiaryID** | **ProjectID** |

**Links beneficiaries to projects, normalized to avoid redundant relationships.**

**Beneficiary Service Table: (Provided)**

|  |  |  |
| --- | --- | --- |
| **BeneficiaryServiceID** | **BeneficiaryID** | **ServiceID** |

**Links beneficiaries to services provided to them (BeneficiaryServiceID, ServiceID).**

**Service Table**

|  |  |
| --- | --- |
| **ServiceID** | **ServiceDetail** |

**Stores service details independently, enabling reuse and consistent updates.**

**Location to Resource: (Located at)**

|  |  |  |
| --- | --- | --- |
| **ResourceLocationID** | **LocationID** | **ResourceID** |

**Maps resources to locations, allowing flexibility in resource allocation.**

**Partnership Details Table**

|  |  |
| --- | --- |
| **PartnershipID** | **PartnershipDetail** |

**Captures additional partnership attributes without embedding them in the main partnership table.**