****

**Lebanese American University**

**Department of Computer Sciences and Mathematics**

**Course: CSC/BIF 375**

**Group Name: Dedicate Keys**



**Group Leader:**

**Mohammad Mzayik**

**Group Members:**

**Ahmad Kashmar, Ali Mhanna, Mohamed Awad, Mohammad Mzayik**

**Submission Date: 28/11/2024**

Contents

[Phase 1 The ER Diagram: 3](#_Toc184556779)

[Introduction: 3](#_Toc184556780)

[Legend: 5](#_Toc184556781)

[Diagram 6](#_Toc184556782)

[Description of each entity type: 7](#_Toc184556783)

[Description of the relationships between the entity types: 20](#_Toc184556784)

[Phase 2 ER to Relational Mapping Algorithm: 30](#_Toc184556785)

[Step 1: Mapping of Regular Entity Types: 30](#_Toc184556786)

[Step 2: Mapping of Weak Entity Types: 31](#_Toc184556787)

[Step 3: Mapping of Binary 1:1 Relationship Types 32](#_Toc184556788)

[Step 4: Mapping of Binary 1:N Relationship Types 32](#_Toc184556789)

[Step 5: Mapping of M:N Relationship Types 33](#_Toc184556790)

[Step 6: Mapping of Multivalued Attributes 34](#_Toc184556791)

[Step 7: Mapping of N-ary Relationships 35](#_Toc184556792)

[Phase 3: Database Implementation 36](#_Toc184556793)

[Table Creation 36](#_Toc184556794)

[Insertion of Data into the Database Tables 51](#_Toc184556795)

[Queries 75](#_Toc184556796)

[Phase 4: Normalization for Relational Databases 84](#_Toc184556797)

[**1. Donor Table** 85](#_Toc184556798)

[**2. Campaign Table** 86](#_Toc184556799)

[**3. Donation Table** 87](#_Toc184556800)

[**4. Volunteer Table** 88](#_Toc184556801)

[**5. Patient Table** 89](#_Toc184556802)

[**6. Event Table** 89](#_Toc184556803)

[7. **Emergency\_Request Table** 90](#_Toc184556804)

[**8. Training\_Program Table** 91](#_Toc184556805)

[**9. Disaster\_Event Table** 92](#_Toc184556806)

[**10. Feedback Table** 92](#_Toc184556807)

[**11. Organization Table** 93](#_Toc184556808)

[**12. Inventory\_Management Table** 93](#_Toc184556809)

[**13. Medical\_Supply Table** 94](#_Toc184556810)

[**14. Volunteer\_Schedule Table** 95](#_Toc184556811)

[**15. Volunteers\_For Table** 95](#_Toc184556812)

[**16. Partners\_With Table** 96](#_Toc184556813)

[**17. Requires Table** 96](#_Toc184556814)

[**18. Requests Table** 98](#_Toc184556815)

[**19. Assigned\_For Table** 98](#_Toc184556816)

[**20. Participates\_In Table** 99](#_Toc184556817)

[**21. Participants Table** 99](#_Toc184556818)

# Phase 1 The ER Diagram:

## Introduction:

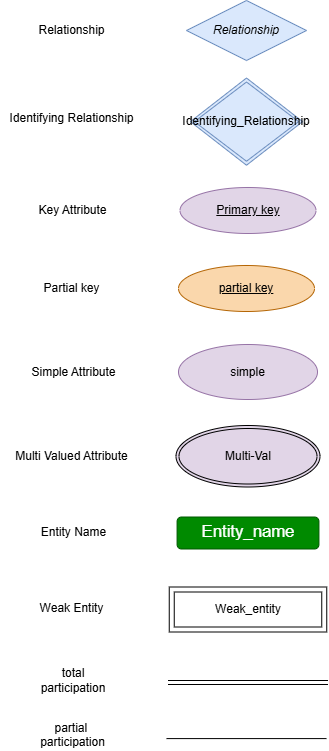
**The Red Cross is an organization always at the ready to save a life and promote good health and well-being. They play a crucial, indispensable role in society, especially now during times of war. However, we cannot give enough credit to the scale of missions the Red Cross needs to manage. It is enough to say they are needed all over the country at all times, and that yes, it is a life or death situation in most cases.**

**Thus, our collaborative team comprised of the members Ali, Mohammad M., Mohammed A., Ahmad, and Lara; also known as the ‘Dedicate Keys’, has been set on the mission to design an efficient database for the Red Cross. Our name is symbolic and references our – and the Red Cross- ‘s dedication to saving lives. On the other hand, it also references ‘candidate keys’ which exist in databases – our primary focus.**

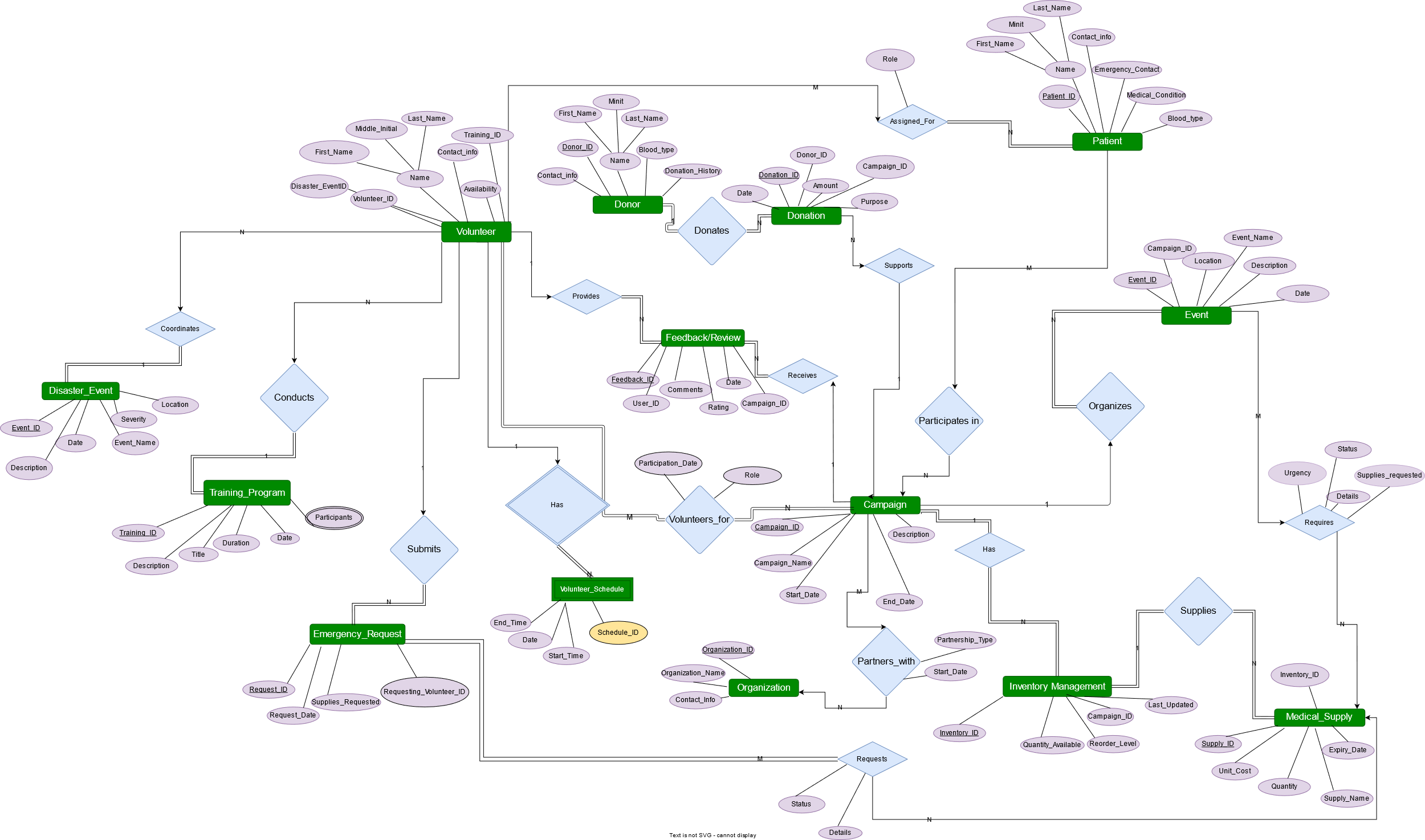
**The initial phase of our project consists of an Entity-Relationship (ER) diagram we have worked on and optimized. This ER diagram represents all the participating entities; tables we deem crucial to have in our database, where we will classify information. The diagram illustrates these entities, along with relationships they are involved in, to further clarify why each entity is crucial to exist as a table of its own, with its own attributes. Lastly, this diagram puts us face-to-face with a model of our database that helps visualize any redundancy or omitted data that would otherwise be important to have. So, with our ER diagram, we aim to model our future database with all its entities, attributes, relationships, and constraints; trying to make it as optimal of a model as possible.**

**In our second phase, we hope to map our ER diagram into a logical structure. We go upon this mission following the seven-step mapping algorithm which involves: the mapping of regular entity types, weak entity types, the binary 1:1, 1:N, and M:N relationship types, multivalued attributes, and lastly the mapping of N-ary relationship types.**

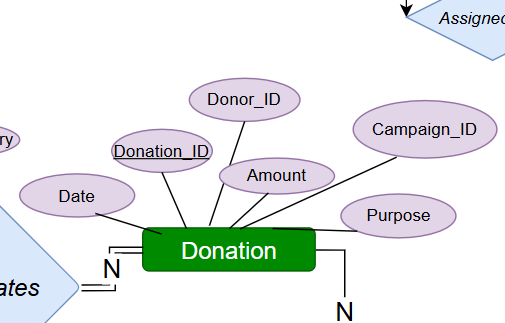
## Legend:

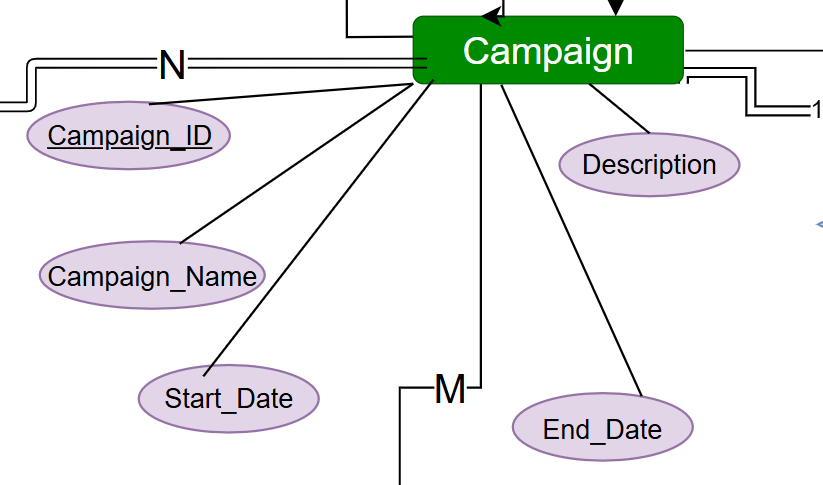
****

## Diagram

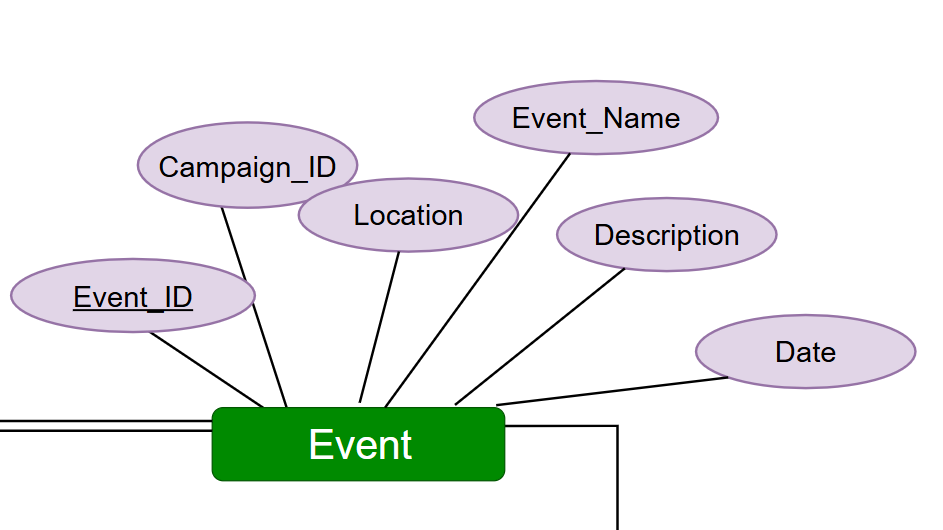


## Description of each entity type:

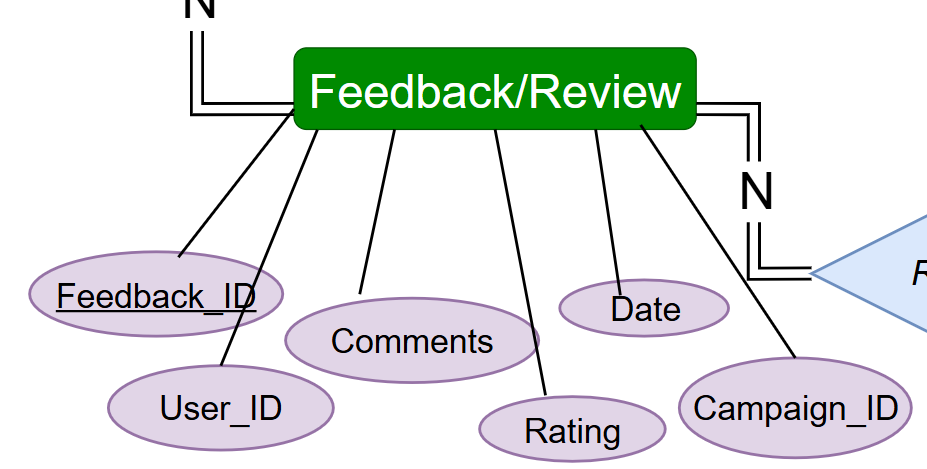
1. **Donors**: Donors have the attributes Donor\_ID, Name, Blood\_Type, Donation\_History, and Contanct\_info.
2. Donor\_ID: it is the identifying (primary key) attribute of this Donor entity, mainly used to destinguish a donor from all other donors since this ID will always be unique.
3. Name: Name is a composite attribute that holds the Donor’s First name, middle initial, and last name.
4. Blood\_Type: For blood donations, it is crucial to identify the donor’s blood type to assess their compatibility with the recipient – which is a matter of life and death.
5. Donation\_History: the donation history of a donor which would be the date of their last donation, would give an implication on their eligibility to donate at the moment.
6. Contact\_Info: Lastly, this attribute would hold the donor’s phone number so that they can be contacted when they are needed or to inform them about any defects in their donated blood sample; which is crucial for their own health and well-being.
7. **Donation**: Donations possess the attributes Donation\_ID, Donor\_ID, Campaign\_ID, Date, Amount, and Purpose.
8. Donation\_ID: this is the primary key of the Donation entity, it serves as a unique identifier of every donation in posession of the Red Cross.
9. Donot\_ID: This is the donor’s unique identifier, crucial in this Donation entity since it helps trace back the source of every donation. This is crucial for obvious reasons such as determining the blood type.
10. Campaign\_ID: This is the campaign’s unique identifier; the campaign which this donation will be supporting.
11. Date: This attribute identifies when the donation was made, and it is crucial for many legal, financial, and administrative reasons.



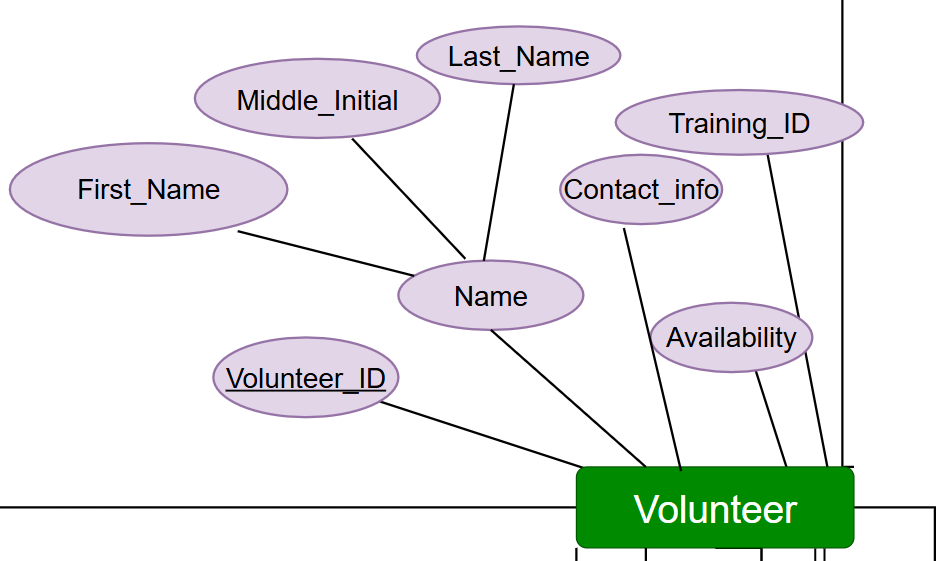
1. **Campaigns**: Campaigns have the attributes ‘Campaign\_ID’, ‘Campaign\_Name’, ‘Start\_Date’, ‘End\_Time’, and Description’.
2. Campaign\_ID: primary key for campaign entity type, and each campaign has a unique ID.
3. Campain\_Name: name or title given to the campaign. Used to differentiate campaigns
4. Description: A detailed description of the campaign's purpose, objectives, or target audience.
5. Start\_Date: The date when the campaign is scheduled to start or when it started.
6. End\_Date: The date when the campaign is scheduled to end. Used to keep track of the duration of every campaign. Important for administrative work.



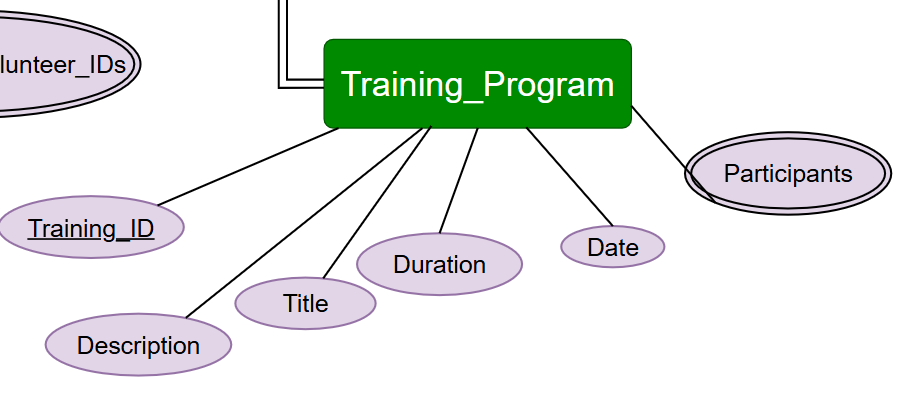
1. **Events**: Events have The attributes ‘Event\_ID’, ‘Campaign\_ID’, ‘Event\_Name’, ‘Date’, ‘Location’, and ‘Description’.
2. Event\_ID: this is the primary key of the Event entity; it serves as a unique identifier of every event of the Red Cross.
3. Campaign\_ID: this is a foreign key; this belongs to the campaign entity type.
4. Location: determines where this event will take place. Important for volunteers of event and people who attend.
5. Event\_Name: the name or title given for this event. Used to distinguish events.
6. Description: A detailed description of the event’s purpose, objectives, or target audience.
7. Date: specifies when the event is set to take place or when it did.



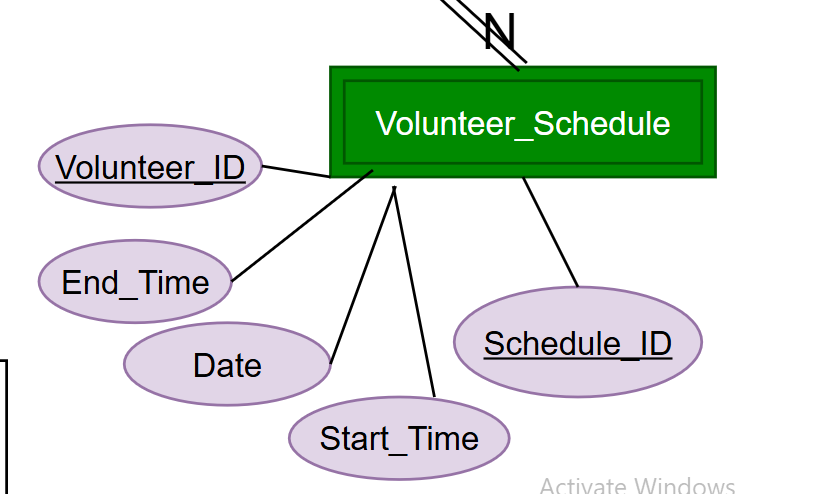
1. **Feeback**: The Feedback entity type has the attributes ‘Feedback\_ID’, ‘User\_ID’, ‘Comments’, ‘Rating’, and ‘Date’.
2. Feedback\_ID: primary key of this entity, serves as a unique identifier for the feedback given.
3. User\_ID: foreign key, serves as an identification key for the user who submitted the feedback.
4. Comments: the comments given under the feedback sent. Fully taken into consideration in order to improve the campaign.
5. Rating: a score given to the campaign on a scale to determine how much the user benefitted from the campaign.
6. Date: indicates the date the feedback was given.
7. Campaign\_ID: foreign key, belongs to campaign. Used to identify which campaign was given the specific feedback or review.



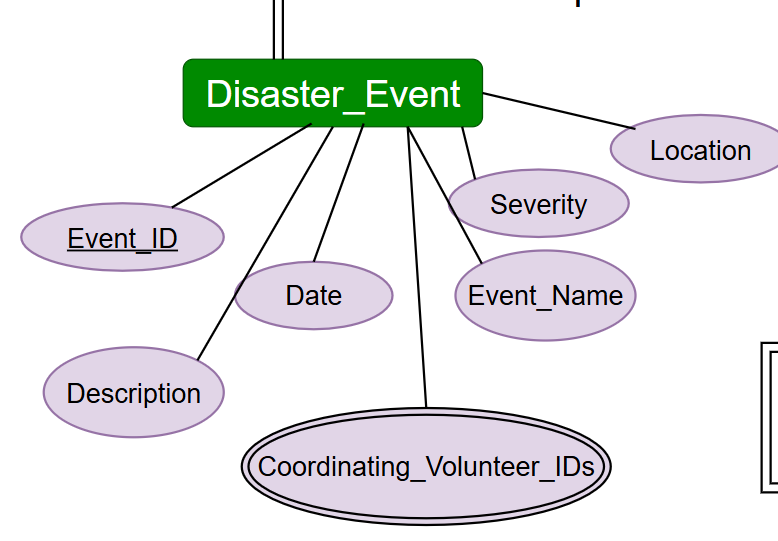
1. **Volunteers**: Volunteers are identified by the attributes Volunteer\_ID, Name, Contact\_info, Availability, Training\_ID.
2. Volunteer\_ID: primary key for this entity, used as a unique identification for the volunteers.
3. Name: Name is a composite attribute that holds the Donor’s First name, middle initial, and last name.
4. Contact\_info: this attribute would hold the donor’s phone number so that they can be contacted when they are needed.
5. Availability: key for knowing when this volunteer is available incase of a medical emergency.
6. Training\_ID: foreign key; an identification for what training this volunteer belongs to. Important for security measures and organization purposes.



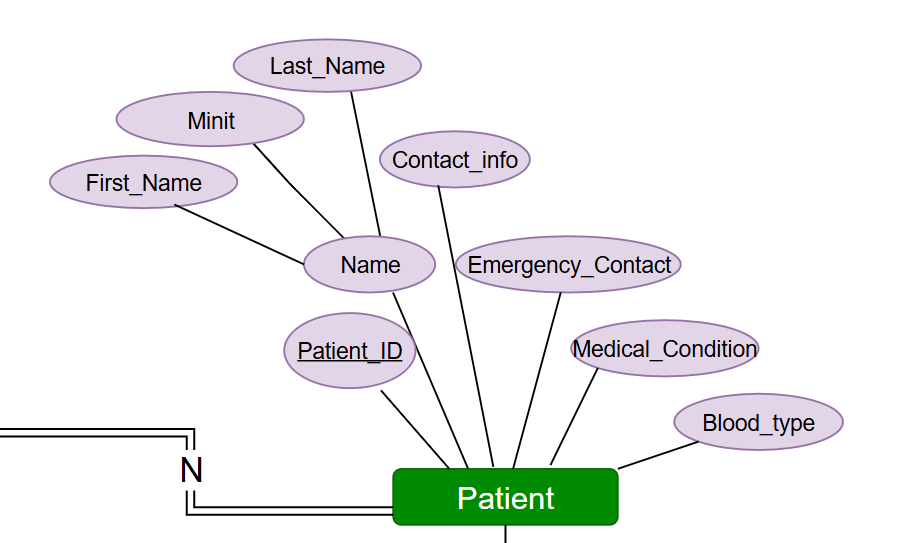
1. **Training\_Program**: this entity has the attributes ‘Training\_ID’, ‘Title’, ‘Description’, ‘Date’, ‘Duration’, and ‘Participants’
2. Training\_ID: primary key of this entity. A unique identification for distinction of the training program.
3. Description: a descriptive text that states what this program’s objectives and purpose.
4. Title: a name for the training program given by the organisers.
5. Duration: how long this training program will take. Important for participants for them to check for time conflicts.
6. Date: date for when this training program began.
7. Participants: a multivalued key because many participants can attend one training program.



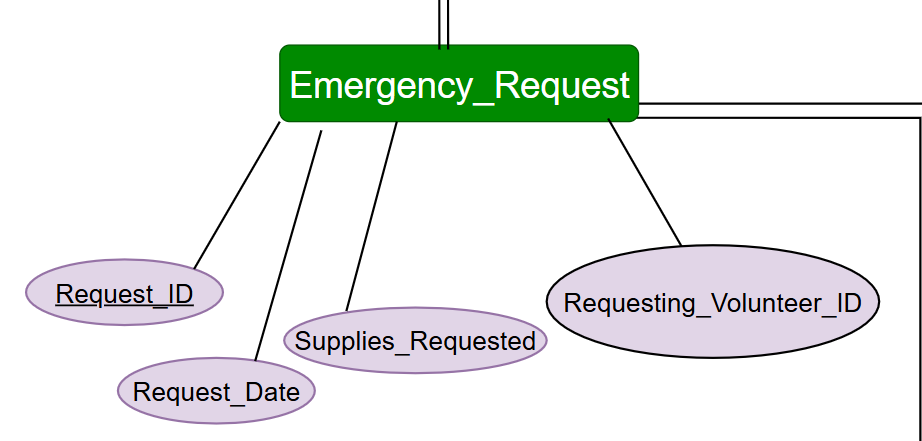
1. **Volunteer\_Schedule**: This is a weak entity and has the attributes ‘Schedule\_ID’, ‘Volunteer\_ID’, ‘Date’, ‘Start\_Time’, and ‘End\_Time’.
2. {Schedule\_ID, Volunteer\_ID}: primary key; the identification of the volunteer that set this schedule and the identification of the schedule that was put.
3. Date: date that tells when this schedule was put.
4. Start\_Time: date that shows when the schedule shall begin.
5. End\_Time: date that shows when the schedule ends.



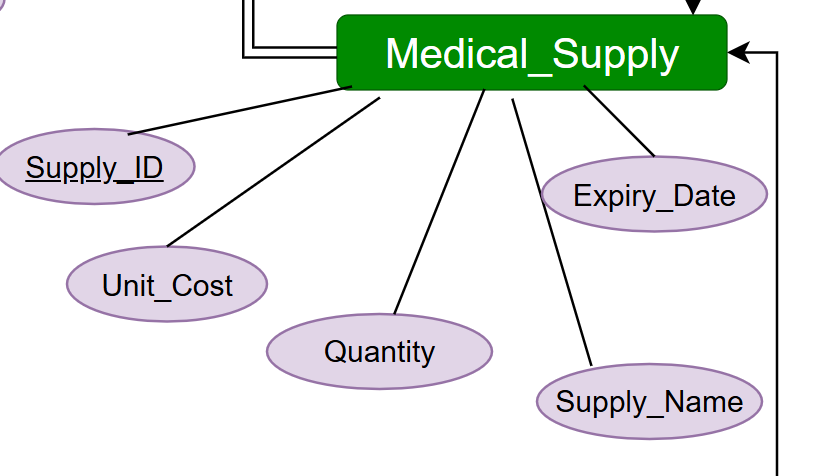
1. **Disaster\_Event**: Here we observe the attributes ‘Event\_ID’ the primary key, as well as ‘Description’, ‘Date’, ‘Coordinating\_Volunteer\_ID’, ‘Event\_Name’, ‘Severity’, and ‘Location’.
2. Event\_ID: primary key that is a unique identification number for the disaster that took place.
3. Description; a detailed description on what happened or what led to this disaster event.
4. Date: date for when this event took place.
5. Coordinating\_Volunteer\_ID: this is a multivalued key; an identification number for the volunteer/s that coordinated this disaster event. Multiple volunteers can coordinate this event at once.
6. Event\_Name: name or title for this event.
7. Severity: This attribute is important especially in the case where there are multiple disaster events where one requires significantly more attention than others.
8. Location: a setting for where this event occurred.



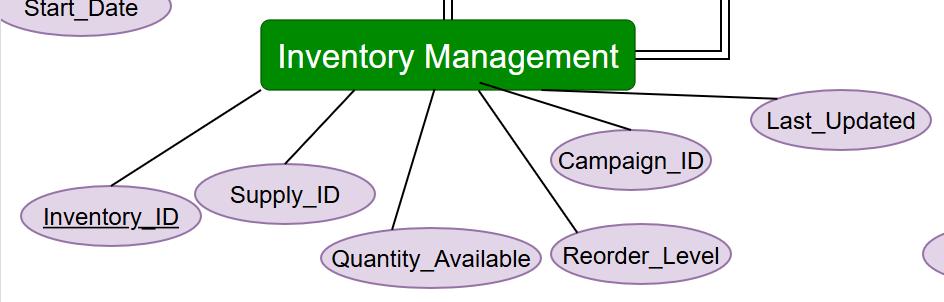
1. **Patient**: Similarly, Patients have the primary key Patient\_ID and the attributes ‘Name’, ‘Contact\_info’, ‘Medical\_Condition’, ‘Emergency\_Contact’, and ‘Blood\_type’.
2. Patient\_ID: primary key of this entity, a unique identification number for each patient.
3. Name: Name is a composite attribute that holds the patients’s First name, middle initial, and last name.
4. Contact\_info: this includes his phone number in case of an emergency.
5. Medical\_Condition: any health conditions the patients suffers from so he can be treated accordingly.
6. Blood\_Type: Also in case of emergencies, if the patient suffers blood loss.
7. Emergency\_Contact: A relative’s phone number, or a close friend’s contact, so they can be called in case of an emergency.



1. **Emergency\_Request**: This entity is created to handle emergency requests because they deserve their own undivided attention. It has the attributes ‘Request\_ID’, ‘Patient\_ID’ ,Volunteer\_ID’, ‘Supplies\_Requested’, ‘Details’, ‘Status’, and ‘Request\_Date’ .
2. Request\_ID: the primary key of this entity, it serves as a unique identification number for each request.
3. Requesting\_Volunteer\_ID: this is a foreign key. Incase if a volunteer submits the request.
4. Supplies\_Requested: sometimes an emergency is required to have certain supplies to deal with it.
5. Request\_Date: a date for when the request was submitted.

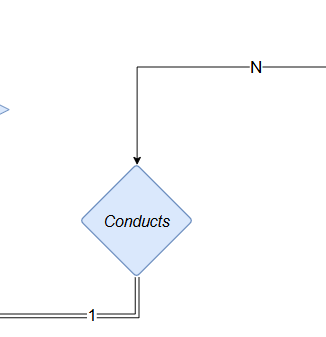


1. **Medical\_Supply**: The entity Medical\_Supply has the attributes ‘Supply\_ID’ its primary key, in addition to ‘Supply\_Name’, ‘Quantity’, ‘Unit\_Cost’, and ‘Expiry\_Date’.
2. Supply\_ID: this is the primary key of this entity. Its an identification number for each medical supply.
3. Supply\_Name: a name or title given for every medical supply.
4. Quantity: the amount of the supply. Important for storing matters.
5. Unit\_Cost: how much this supply costed. Important for running finances.
6. Expiry\_Date: each supply unit has an expiry date, to know when this supply is good before.

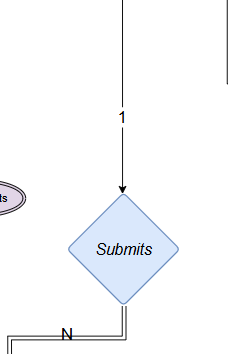


1. ‘**Inventory\_Management’**: Lastly, this entity has the primary key ‘Inventory\_ID’, it has the foreign key ‘Supply\_ID’ as well as the attributes ‘Quantity\_Available’, ‘Reorder\_Level’, ‘Campaign\_ID’ and ‘Last\_Updated’
2. Inventory\_ID: primary key of this entity. Serves as a unique identification number for each inventory.
3. Supply\_ID: foreign key for this entity, in case of a supply requested.
4. Quantity\_Available: this key is to know how much of a certain supply is left.
5. Reorder\_Level: this is to reorder the inventory.
6. Campaign\_ID: also, a foreign key. When a campaign needs a supply, its ID is required.
7. Last\_Updated: this is important to know if when was the last time the inventory was restocked.

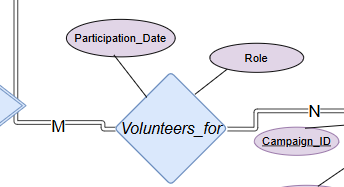
## Description of the relationships between the entity types:



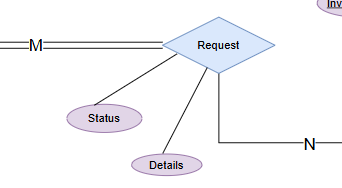
**1: Conducts:** The relationship between the volunteer entity type and the training program entity type is N:1. The training program shows total participation, while the volunteer entity type shows partial participation because all training programs should be conducted by volunteers, but a volunteer may not be involved in a training program. Moreover, a training program may be conducted by more than 1 volunteer, but a volunteer can only conduct one training program. This helps them focus well on their assigned responsibilities in the program they manage. This is important because a training program may involve showcasing how to perform emergency operations or deal with emergency situations… Thus, the volunteer’s full attention should be on the sole program they conduct.



**2: Submits:** This is a 1:N binary relationship between the “Volunteer” entity and the “Emergency\_Request” entity. Volunteer is a partial participant here since not all volunteers will be submitting emergency requests. On the contrary, no emergency request exists on its own, it has to be submitted by a volunteer; which is why this entity is a total participant in this relationship.

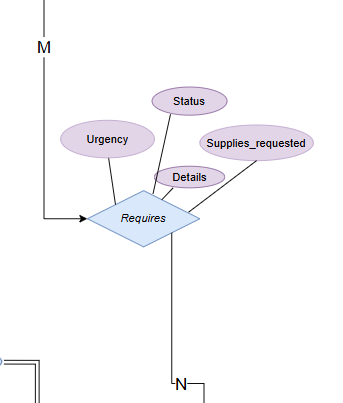


**3: Volunteers\_for:** This is an M:N binary relationship where multiple volunteers get to volunteer for multiple campaigns, and each campaign can have multiple volunteers supporting it. Both entities have mandatory participation since no campaign is without volunteers, and every volunteer is ‘volunteering for’ a campaign – else they would not even be a volunteer! It has the attributes ‘role’ to indicate the role of each volunteer in the campaign as well as the attribute ‘participation\_date’ to indicate when this volunteer began participating in the campaign.



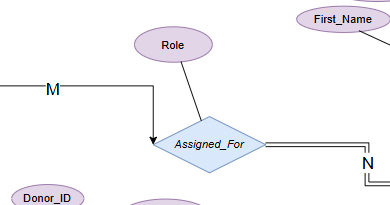
**4: Request:** This is an M:N binary relationship where multiple emergency requests can be requesting or asking for multiple medical supplies. Here, the participation of Emergency\_Request is mandatory since it is literally a request in itself; meaning, it has to actually go and request the supplies and not just exist as a request. On the other hand, medical supplies can exist without being requested, which is why this entity is a partial participant.

This relationship also has the attributes Status and Details which indicate whether the request has been sent, processed, shipped... The details of the request indicate specific requirements like a comment explaining the purpose, urgency, or maybe a deadline for the request to be shipped.



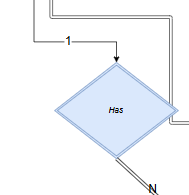
**5: Requires:** This relationship is between Event and Medical\_Supply entities. It is an M:N binary relationship where many events require a variety of medical supplies. Both entities are partial participants. Not all supplies are required by events, and not all events will necessarily require medical supplies; it could be just an informative event for example or a fundraiser...

**This relationship has the entities Urgency (how important the request is), Status (submitted, processed, fulfilled…), Supplies\_requested (Testing tools, samples, solvents, medication, bandages…) and Details which provides any additional information the client may want to add.**

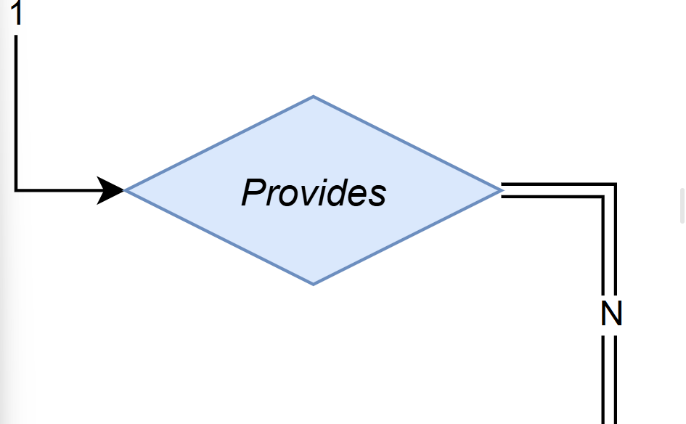


**6: Assigned\_For:** The Assigned\_For relationship links a volunteer to their assigned patient. It is important for both parties to know who they are dealing with as well as forwhoever is managing the database to know, in case anything goes wrong or a reassignment is necessary. A volunteer will not necessarily be treating a patient. Thus, they are partial participants in this relation. However, every patient will be handled by a volunteer, which is why they are total participants.

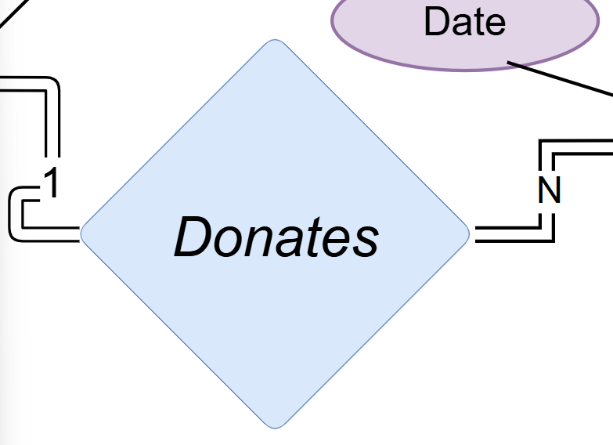
This relationship has the attribute ‘role’ which indicates the role of the volunteer in this assignment; are they their primcary caregiver, medical assistant, companion, translator, resource provider (providing food and medication)?...

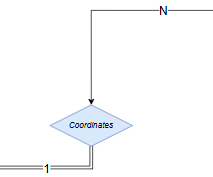


**7: Has:** This is an identifying 1:N relationship linking volunteers to their schedule because every volunteer must have a schedule or multiple schedules. However, the schedule – which is the only total participant here – would not exist had there been no volunteer.

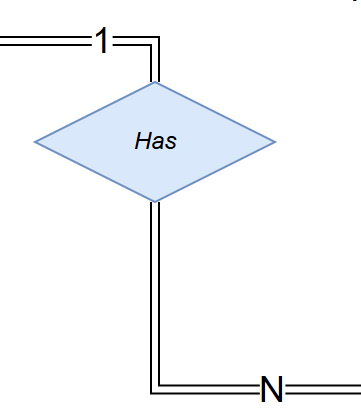


**8: Provides**: This is a 1:N relationship showcasing how one volunteer can submit multiple feedbacks or reviews. Every feedback has to come from a volunteer. However, not every volunteer has to provide feedback, which shows with the partial participation for the volunteers. On the other hand, no feedback exists on its own, it will always be submitted by a volunteer, hence the total participation for the feedback/review entity.

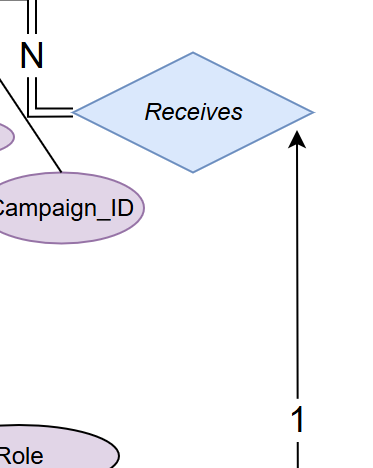
****

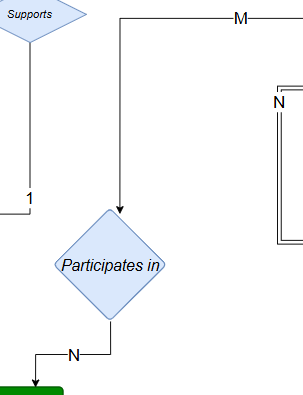
**9: Donates:** This relationship is a 1:N binary relationship linking the donor with their respective donation or multiple donations. Both entity types have total participation, since every donor has a donation and every donation comes from a donor.

**10: Coordinates:** This N:1 binary relationship reveals how a disaster event can be coordinated by one or multiple volunteers, however only the “disaster\_event” entity shows total participation since not every volunteer has to be coordinating a disaster event but every disaster event has to be conducted by at least one volunteer. Since disaster events are crucial for a volunteer to focus their efforts on, we do not allow them to conduct multiple events. But since collaborative efforts are important for a disaster event (a fire, a collapsed building,…) multiple volunteers are allowed to be involved per event.

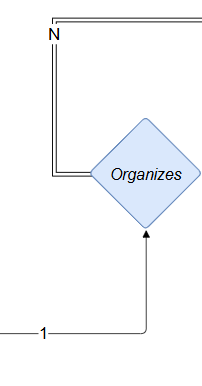
****

**11: Has:** The Has relationship is a 1:N binary relationship indicating how every campaign can have one or many inventories for their medical supplies. However, an inventory can only belong to one campaign. This is important for confidentiality and security reasons. Both entities are total participants since a campaign should have at least one inventory for its medical supplies, and an inventory has to belong to a campaign.

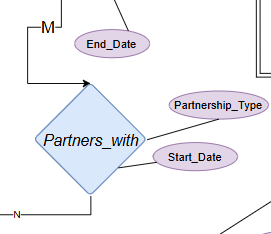
****

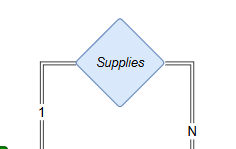
**12: Receives:** This relationship permits a campaign to receive one or many feedbacks/reviews with its 1:N binary nature. A campaign receiving feedback is optional – hence the partial participation. However, a feedback must be submitted for a certain campaign – so it would be a total participant in this relation.

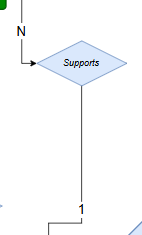
**13: Participates in:** In the red cross’s campaign, this M:N relationship allows multiple patients to be involved in multiple campaigns. Participation is optional for both entities; a campaign can involve no patients, and a patient may not be involved in any campaign.

****

**14: Organizes:** This is a 1:N binary relationship where a campaign is allowed to organize one or more events. The event entity type shows total participation while the campaign entity type indicates partial participation, since every event has to be organized by a campaign however not every campaign has to organize an event.

****

**15: Partners with:** This M:N relationship indicates that many campaigns are allowed to partner up with several partnerships and vice versa. The two entity types have partial participation since a campaign may be collaborating with no one, and an organization can ****exist without partnering with any campaign.

**16: Supplies:** This relationship is a 1:N binary relationship between inventory management and Medical\_supply entities. Both participants show mandatory participation. No inventory management will withhold its medical supplies, and no supply will come from any other source than the existing inventories. Here, one inventory management can supply multiple tools, medicines, products… However, a certain supply batch will obviously come from only one source.

**17: Supports:** Lastly, the relationship Supports is an N:1 binary relationship where participation for both entities is optional. A donation may or may not support a campaign. A campaign may not receive any donation if it does not need them. Many donations can support a campaign, but a donation can never be split for multiple campaigns – it is a blood donation so that won’t make sense; there are many constraints like the amount of blood donated and fear of contamination when splitting a sample.

# Phase 2 ER to Relational Mapping Algorithm:

## Step 1: Mapping of Regular Entity Types:

In this step, we map the regular entity types – each as its own table with its primary key underlined. These strong entity types in our Dedicate Keys database are:

Donor

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| First\_Name | Last\_Name | Middle\_Initial | Conact\_info | Blood\_type | Donation\_History | Donor\_ID |

The DONOR entity contains simple and composite attributes. This relation includes all simple attributes and the primary key Donor\_ID is underlined. The DONOR entity has Name as a composite attribute which includes the simple attributes First\_Name, Minit, and Last\_Name

Donation

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Donation\_ID | Amount | Purpose |

The DONATION entity contains all simple attributes and the primary key Donation\_ID is underlined.

Volunteer

|  |  |  |  |
| --- | --- | --- | --- |
| First\_Name | Last\_Name | Middle\_Inital | Volunteer\_ID |
| Contact\_info | Availability | Volunteer\_interest |

The VOLUNTEER entity contains simple and composite attributes. This relation includes all simple attributes and the primary key Volunteer\_ID which is underlined. The PATIENT entity has Name as a composite attribute which includes the simple attributes First\_Name, Middle\_Initial, and Last\_Name.

Patient

|  |  |  |  |
| --- | --- | --- | --- |
| First\_Name | Last\_Name | Middle\_Inital | Patient\_ID |
| Medical\_condition | Blood\_type | Emergency\_Contact | Contact\_info |

The PATIENT entity contains simple and composite attributes. This relation includes all simple attributes and the primary key Patient\_ID is underlined. The PATIENT entity has Name as a composite attribute which includes the simple attributes First\_Name, Minit, and Last\_Name

Event

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Event\_Name | Location | Event\_ID | Description | Date |

The EVENT entity contains all simple attributes. This relation includes all simple attributes and the primary key Event\_ID is underlined

Emergency\_Request

|  |  |  |
| --- | --- | --- |
| Request\_ID | Request\_Date | Supplies\_Requested |

The Emergency\_Request entity contains all simple attributes. This relation includes all simple attributes and the primary key Request\_ID is underlined.

Training\_Program

|  |  |  |
| --- | --- | --- |
| Training\_ID | Description | Title |
| Participants | Date | Duration |

The TRAINING\_PROGRAM entity contains simple and multivalued attributes. This relation includes all simple attributes and the primary key training\_ID is underlined. The TRAINING\_PROGRAM entity has Participants as a multivalued attribute which is included in this relation.

Disaster\_Event

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Event\_ID | Description | Date | Severity | Location | Event\_Name |

The DISASTER\_EVENT entity contains all simple attributes. This relation includes all simple attributes and the primary key Event\_ID is underlined.

Feedback/Review

|  |  |  |  |
| --- | --- | --- | --- |
| Feedback\_ID | Comments | Rating | Date |

The FEEDBACK/REVIEW entity contains all simple attributes. This relation includes all simple attributes and the primary key Feedback\_ID is underlined.

Campaign

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Campaign\_ID | Campaign\_Name | Start\_Date | End\_Date | Description |

The CAMPAIGN entity contains all simple attributes. This relation includes all simple attributes and the primary key Campaign\_ID is underlined.

Organization

|  |  |  |
| --- | --- | --- |
| Organization\_ID | Organization\_Name | Contact\_Info |

The ORGANIZATION entity contains all simple attributes. This relation includes all simple attributes and the primary key Organization \_ID is underlined

Inventory Management

|  |  |  |  |
| --- | --- | --- | --- |
| Inventory\_ID | Quantity\_Available | Reorder\_Level | Last\_Updated |

The INVENTORY\_MANAGEMENT entity contains all simple attributes. This relation includes all simple attributes and the primary key Inventory \_ID is underlined

Medical Supply

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Supply\_ID | Unit\_Cost | Quantity | Supply\_Name | Expiry\_Date |

The MEDICAL\_SUPPLY entity contains all simple attributes. This relation includes all simple attributes and the primary key Supply \_ID is underlined

## Step 2: Mapping of Weak Entity Types:

In this step we map the only weak entity we have which is VOLUNTEER\_SCHEDULE. We include only the simple attributes. Moreover, this entity possesses a foreign key which references its owner entity. The combination of this foreign key and the partial key of this entity make up its primary key.

Volunteer\_Schedule (weak)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Volunteer\_ID | End\_Time | Date | Start\_Time | Schedule\_ID |

In this weak entity, the foreign key ‘Volunteer\_ID’ references its owner entity of type VOLUNTEER. This entity also has the simple attributes End\_time, Date, Start\_time, and Schedule\_ID. It is worthy to note that Schedule\_ID here is a partial key, and combined with Volunteer\_ID the foreign key, they both make up the primary key of this weak entity type. Thus, we underline them both.

## Step 3: Mapping of Binary 1:1 Relationship Types

This step can be done in 3 ways. The first approach would be the foreign key approach where we choose the entity on the total participation side and add to it as foreign key the primary key of the other. A second approach would be the merge relation approach where we merge both entities into a single relation. This applies when both are total participants. The third approach is the cross-reference or the relationship relation approach which involves creating a third relation that has the primary key of each participating relationships.

However, our diagram does not seem to involve any 1:1 relationship so we skip this step.

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

In this step, we are mapping 1:N binary relationships. We do that by including a foreign key in the relation on the N side of this 1:N relationship. This foreign key references the primary key of the other participant.

Volunteer (COORDINATES, CONDUCTS)

|  |  |  |  |
| --- | --- | --- | --- |
| First\_Name | Last\_Name | Middle\_Inital | Volunteer\_ID |
| Contact\_info | Availability | Training\_ID | Disaster\_event\_ID |

A Volunteer will COORDINATE ONE Disaster\_Event. Thus, here we include Disaster\_event\_ID which references the primary key of the Disaster event (EVENT\_ID) which the volunteer will coordinate.

A volunteer may conduct one training program, whose primary key ‘Training\_ID’ is referenced here as a foreign key with the same name.

Emergency\_Request (SUBMITS)

|  |  |  |  |
| --- | --- | --- | --- |
| Request\_ID | Request\_Date | Supplies\_Requested | Requesting\_Volunteer\_ID |

Many emergency requests may be submitted by one volunteer. Being on the many side of the relationship, we reference the VOLUNTEER entity type with REQUESTING\_VOLUNTEER\_ID the foreign key referring to its primary key ‘Volunteer\_ID’.

Volunteer\_Schedule (HAS)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Volunteer\_ID | End\_Time | Date | Start\_Time | Schedule\_ID |

This relation does not change. Being on the N side of the ‘VOLUNTEER HAS VOLUNTEER\_SCHEDULE’ relationship, it already has the foreign key ‘Volunteer\_ID’ we would have otherwise added to establish this relationship.

Feedback/Review (PROVIDES, RECEIVES)

|  |  |  |
| --- | --- | --- |
| Feedback\_ID | User\_ID | Comments |
| Rating | Date | Campaign\_ID |

In this relation, we add the foreign key ‘Volunteer\_ID’ to reference the VOLUNTEER which PROVIDES a REVIEW. We also add the foreign key CAMPAIGN\_ID to reference the CAMPAIGN which RECEIVES the REVIEW. This is because the Feedback/Review relation is on the N side of both relationships.

Medical Supply (SUPPLIES)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Supply\_ID | Unit\_Cost | Quantity | Supply\_Name | Expiry\_Date | Inventory\_ID |

This relation is on the many side of the relationship SUPPLIES with entity type INVENTORY MANAGEMENT. Thus, we reference this entity’s primary key with the foreign key INVENTORY\_ID.

Event (ORGANIZES)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Event\_Name | Campaign\_ID | Location | Event\_ID | Description | Date |

This relation is on the many side of the relationship CAMPAIGN ORGANIZES EVENT. Thus, we reference CAMPAIGN’s primary key with a foreign key in this relation having the same name CAMPAIGN\_ID.

Donation (DONATES, SUPPORTS)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Date | Donation\_ID | Donor\_ID | Amount | Campaign\_ID | Purpose |

This relation is on the many side of the relationship DONOR DONATES DONATION. So, we include a foreign key DONOR\_ID to reference the primary key of DONOR. It is also on the N side of the relationship DONATION SUPPORTS CAMPAIGN. This is why we add another foreign key CAMPAIGN\_ID which references the primary key of CAMPAIGN.

## Step 5: Mapping of M:N Relationship Types

In this step we create a relation for every M:N relationship. In this relation, we include as foreign keys the primary keys of participating entities, as well as any other simple attributes that may be involved. We underline both foreign keys as they constitute the primary key of this relation type.

Volunteers\_for

|  |  |  |  |
| --- | --- | --- | --- |
| Participation\_Date | Role | Volunteer\_ID | Campaign\_ID |

VOLUNTEER VOLUNTEERS FOR CAMPAIGN is a many to many relationship. This is why we add this joining table which has the foreign keys VOLUNTEER\_ID and CAMPAIGN\_ID referencing the primary key (with the same name) of the respective entities. We also add the simple attributes Participation\_Date and Role.

Partners\_with

|  |  |  |  |
| --- | --- | --- | --- |
| Start\_Date | Partnership\_Type | Organization\_ID | Campaign\_ID |

This many to many relationship links CAMPAIGN to ORGANIZATION. Thus, we include foreign keys referencing each entity’s primary key with the same name. Additionally, we add the simple attributes Start\_Date and Partnership\_Type.

Requires

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Urgency | Status | Details | Supplies\_requested | Supply\_ID | Event\_ID |

EVENT REQUIRES MEDICAL SUPPLY is a many to many relationship which we map into this relation. Along with the foreign keys SUPPLY\_ID and EVENT\_ID, referencing the primary keys of Medical\_Supply and EVENT respectively, we include the simple attributes Urgency, Status, Details, and Supplies\_Requested.

Requests

|  |  |  |  |
| --- | --- | --- | --- |
| Status | Details | Request\_ID | Supply\_ID |

This many to many relationship is mapped as the relation shown. We add the foreign keys Request\_ID referencing the primary key of EMERGENCY\_REQUEST, as well as Supply\_ID referencing the primary key of MEDICAL\_SUPPLIES. Simple attributes Status and Details are also added.

Assigned\_for

|  |  |  |
| --- | --- | --- |
| Role | Patient\_ID | Volunteer\_ID |

Many volunteers may be assigned to many patients. To keep track of this many to many relationship, we construct this relation which includes foreign keys PATIENT\_ID and VOLUNTEER\_ID referencing the primary keys of the respective entities with the same name. We also add the simple attribute ROLE.

Participates\_in

|  |  |
| --- | --- |
| Campaign\_ID | Patient\_ID |

This many to many relationship PATIENT PARTICIPATES IN CAMPAIGN, is simply mapped into this relation which only has foreign keys CAMPAIGN\_ID and PATIENT\_ID, referring to the primary keys of the respective entities with the same name.

## Step 6: Mapping of Multivalued Attributes

We have only one multivalued attribute which is PARTICIPANTS. For this attribute we will create a separate relation which includes the primary key of the entity it belongs to along with the related attribute. This relation has the foreign key Training\_ID along with the attribute Participant\_Name. The combination of both constitutes the primary key of this multivalued attribute, helping us distinguish between the unique participants in every training program.

Participants

|  |  |
| --- | --- |
| Training\_ID | Participant\_Name |

## Step 7: Mapping of N-ary Relationships

To map N-ary relationships, we would need to create new tables with foreign keys referencing each participating entity along with any extra attributes. However, we do not seem to have any N-ary relationship in our diagram so this step will be skipped.

# Phase 3: Database Implementation

## Table Creation

**1.Donor (Donor\_ID, Name, Blood\_Type, Contact\_Info, Donation\_History):**

**Code:**CREATE TABLE Donor (

Donor\_ID INT PRIMARY KEY,

First\_Name VARCHAR2(50) NOT NULL,

Last\_Name VARCHAR2(50) NOT NULL,

Middle\_Initial CHAR(1),

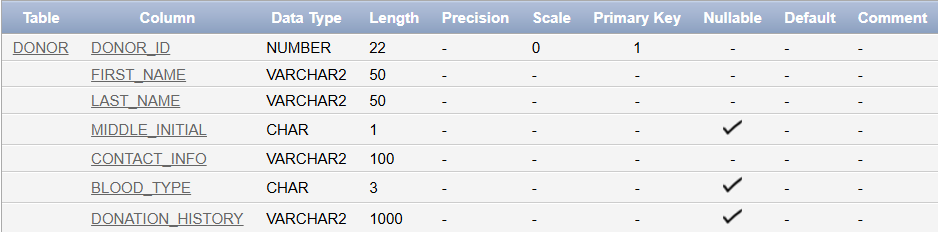
Contact\_Info VARCHAR2(100) NOT NULL,

Blood\_Type CHAR(3) CHECK (Blood\_Type IN ('A+', 'A-', 'B+', 'B-', 'O+', 'O-', 'AB+', 'AB-')),

Donation\_History VARCHAR2(1000)

);

**Explanation:**

The Donor table stores information about individuals who donate blood. Donor\_ID is the primary key, uniquely identifying each donor. Attributes like First\_Name, Last\_Name, and Blood\_Type capture key details, while Donation\_History stores donation dates as a comma-separated string. Constraints like NOT NULL ensure mandatory fields are filled, and CHECK validates blood type values.****

**2. Campaign (Campaign\_ID, Campaign\_Name, Start\_Date, End\_Date, Description)**

**Code:**

CREATE TABLE Campaign (

Campaign\_ID INT PRIMARY KEY,

Campaign\_Name VARCHAR2(100) NOT NULL,

Start\_Date DATE NOT NULL,

End\_Date DATE NOT NULL,

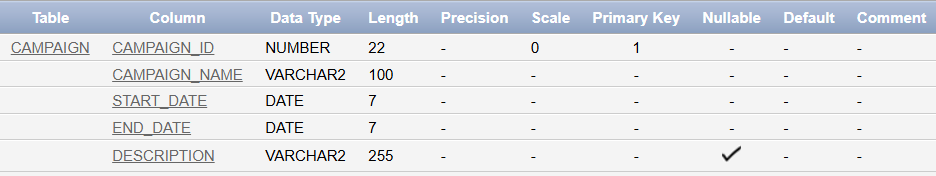
Description VARCHAR2(255),

CONSTRAINT CHK\_Campaign\_Dates CHECK (End\_Date >= Start\_Date)

);

**Explanation:**

The Campaign table manages organized efforts for specific causes. Campaign\_ID is the primary key. Attributes like Campaign\_Name, Start\_Date, and End\_Date describe the campaign. A CHECK constraint ensures the campaign end date is after the start date.

****

**3. Donation (Donation\_ID, Donor\_ID, Campaign\_ID, Donation\_Date, Amount, Purpose)**

**Code:**

CREATE TABLE Donation (

Donation\_ID INT PRIMARY KEY,

Donation\_Date DATE NOT NULL,

Amount NUMBER(10, 2) CHECK (Amount >= 0),

Purpose VARCHAR2(255) NOT NULL,

Donor\_ID INT NOT NULL,

Campaign\_ID INT,

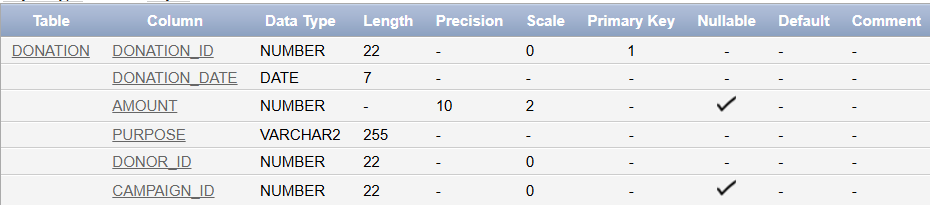
FOREIGN KEY (Donor\_ID) REFERENCES Donor(Donor\_ID),

FOREIGN KEY (Campaign\_ID) REFERENCES Campaign(Campaign\_ID)

);

**Explanation:**

The Donation table records details of each donation. Donation\_ID serves as the primary key. Attributes like Date, Amount, and Purpose provide essential information. Foreign keys link donations to their respective donors (Donor\_ID) and campaigns (Campaign\_ID). Constraints ensure valid donation amounts and mandatory fields.

****

**4. Volunteer (Volunteer\_ID, Name, Contact\_Info, Availability, Volunteer\_Interest, Training\_ID, Disaster\_Event\_ID)**

**Code:**

CREATE TABLE Volunteer (

Volunteer\_ID INT PRIMARY KEY,

First\_Name VARCHAR2(50) NOT NULL,

Last\_Name VARCHAR2(50) NOT NULL,

Middle\_Initial CHAR(1),

Contact\_Info VARCHAR2(100) NOT NULL,

Availability VARCHAR2(255) NOT NULL,

Volunteer\_Interest VARCHAR2(255),

Training\_ID INT,

Disaster\_Event\_ID INT,

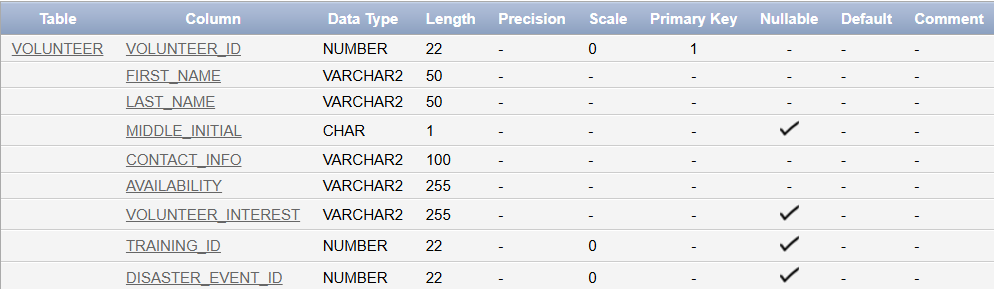
FOREIGN KEY (Training\_ID) REFERENCES Training\_Program(Training\_ID),

FOREIGN KEY (Disaster\_Event\_ID) REFERENCES Disaster\_Event(Event\_ID)

);

**Explanation:**

The Volunteer table manages data about individuals who assist with activities and events. Volunteer\_ID is the primary key. Attributes such as Availability and Volunteer\_Interest capture volunteer preferences. Foreign keys (Training\_ID and Disaster\_Event\_ID) associate volunteers with training programs and disaster events.



**5. Patient (Patient\_ID, Name, Medical\_Condition, Blood\_type, Emergency\_Contact, Contact\_Info)**

**Code:**

CREATE TABLE Patient (

Patient\_ID INT PRIMARY KEY,

First\_Name VARCHAR2(50) NOT NULL,

Last\_Name VARCHAR2(50) NOT NULL,

Middle\_Initial CHAR(1),

Medical\_Condition VARCHAR2(255) NOT NULL,

Blood\_Type CHAR(3) CHECK (Blood\_Type IN ('A+', 'A-', 'B+', 'B-', 'O+', 'O-', 'AB+', 'AB-')),

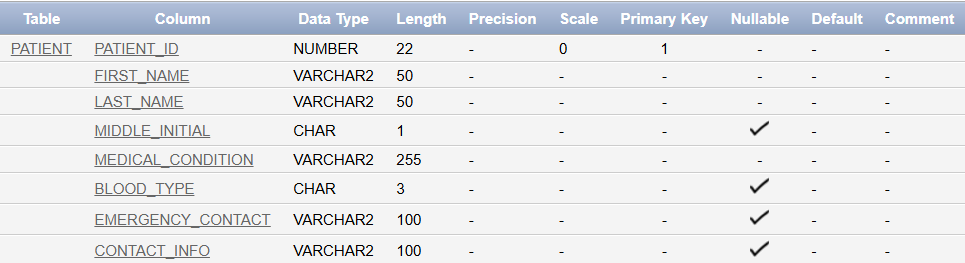
Emergency\_Contact VARCHAR2(100),

Contact\_Info VARCHAR2(100)

);

**Explanation:**

The Patient table stores information about individuals requiring medical attention. Patient\_ID is the primary key. Attributes like Medical\_Condition, Blood\_Type, and Emergency\_Contact provide critical details. Constraints like NOT NULL ensure essential fields are populated, while CHECK enforces valid blood types.



**6. Event (Event\_ID, Event\_Name, Location, Description, Event\_Date, Campaign\_ID)**

**Code:**

CREATE TABLE Event (

Event\_ID INT PRIMARY KEY,

Event\_Name VARCHAR2(100) NOT NULL,

Location VARCHAR2(100) NOT NULL,

Description VARCHAR2(255),

Event\_Date DATE NOT NULL,

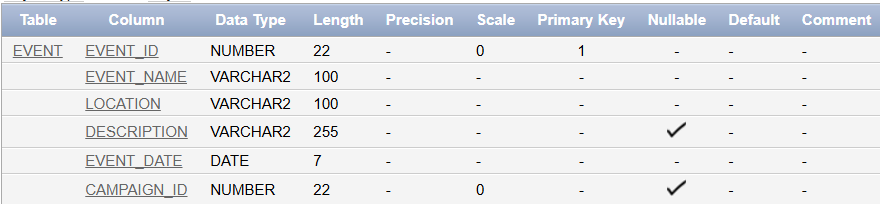
Campaign\_ID INT,

FOREIGN KEY (Campaign\_ID) REFERENCES Campaign(Campaign\_ID)

);

**Explanation:**

The Event table tracks organized events and activities. Event\_ID is the primary key. Attributes such as Event\_Name, Location, and Date describe the event. A foreign key links events to campaigns (Campaign\_ID). This table facilitates planning and management of various events.



**7. Emergency\_Request (Request\_ID, Request\_Date, Supplies\_Requested, Requesting\_Volunteer\_ID)**

**Code:**  
CREATE TABLE Emergency\_Request (

Request\_ID INT PRIMARY KEY,

Request\_Date DATE NOT NULL,

Supplies\_Requested VARCHAR2(255) NOT NULL,

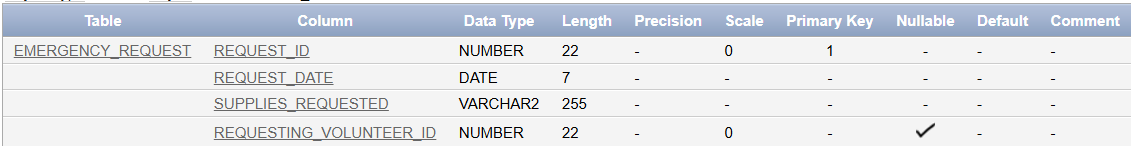
Requesting\_Volunteer\_ID INT,

FOREIGN KEY (Requesting\_Volunteer\_ID) REFERENCES Volunteer(Volunteer\_ID)

);

**Explanation:**

The Emergency\_Request table records urgent supply requests. Request\_ID is the primary key. Attributes like Request\_Date and Supplies\_Requested provide details about each request. A foreign key links requests to the submitting volunteer (Requesting\_Volunteer\_ID).



**8. Training\_Program (Training\_ID, Description, Title, Training\_Date, Duration)**

**Code:**

CREATE TABLE Training\_Program (

Training\_ID INT PRIMARY KEY,

Description VARCHAR2(255) NOT NULL,

Title VARCHAR2(100) NOT NULL,

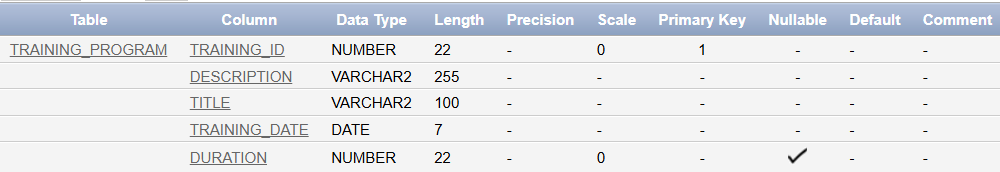
Training\_Date DATE NOT NULL,

Duration INT CHECK (Duration > 0)

);

**Explanation:**

The Training\_Program table manages training session details. Training\_ID is the primary key. Attributes like Title, Description, Training\_Date, and Duration provide essential information. Constraints ensure valid durations and mandatory fields.



**9. Disaster\_Event (Event\_ID, Description, Severity, Location, Event\_Name)**

**Code:**

CREATE TABLE Disaster\_Event (

Event\_ID INT PRIMARY KEY,

Description VARCHAR2(255) NOT NULL,

Disaster\_Date DATE NOT NULL,

Severity VARCHAR2(50) CHECK (Severity IN ('Low', 'Medium', 'High')),

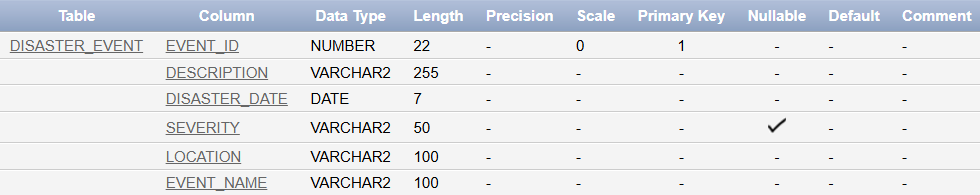
Location VARCHAR2(100) NOT NULL,

Event\_Name VARCHAR2(100) NOT NULL

);

**Explanation:**

The Disaster\_Event table stores information about major incidents. Event\_ID is the primary key. Attributes like Description, Severity, and Location provide details about the disaster. Constraints ensure valid severity levels (e.g., Low, Medium, High).



**10. Feedback (Feedback\_ID, Comments, Rating, Feedback\_Date, Volunteer\_ID, Campaign\_ID)**

**Code:**CREATE TABLE Feedback (

Feedback\_ID INT PRIMARY KEY,

Comments VARCHAR2(1000),

Rating INT CHECK (Rating >= 1 AND Rating <= 5),

Feedback\_Date DATE NOT NULL,

Volunteer\_ID INT NOT NULL,

Campaign\_ID INT NOT NULL,

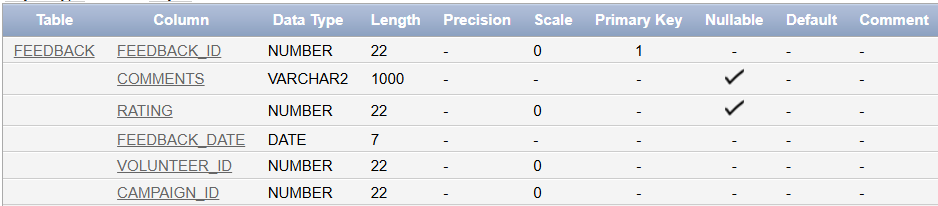
FOREIGN KEY (Volunteer\_ID) REFERENCES Volunteer(Volunteer\_ID),

FOREIGN KEY (Campaign\_ID) REFERENCES Campaign(Campaign\_ID)

);

**Explanation:**

The Feedback table collects reviews about campaigns. Feedback\_ID is the primary key. Attributes like Rating, Comments, and Date store the feedback details. Foreign keys link feedback to the volunteer who provided it (Volunteer\_ID) and the campaign it pertains to (Campaign\_ID).



**11. Organization (Organization\_ID, Organization\_Name, Contact\_Info)**

**Code:**

CREATE TABLE Organization (

Organization\_ID INT PRIMARY KEY,

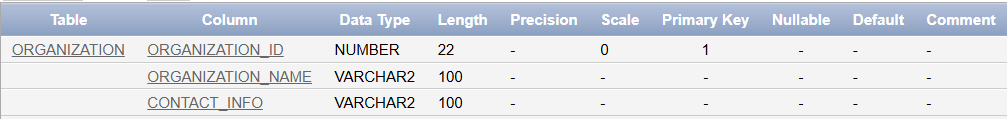
Organization\_Name VARCHAR2(100) NOT NULL,

Contact\_Info VARCHAR2(100) NOT NULL

);

**Explanation:**

The Organization table records information about partner organizations. Organization\_ID is the primary key. Attributes like Organization\_Name and Contact\_Info capture essential details. This table supports collaboration with external entities.



**12. Inventory\_Management (Inventory\_ID, Quantity\_Available, Reorder\_Level, Last\_Updated)**

**Code:**

CREATE TABLE Inventory\_Management (

Inventory\_ID INT PRIMARY KEY,

Quantity\_Available INT CHECK (Quantity\_Available >= 0),

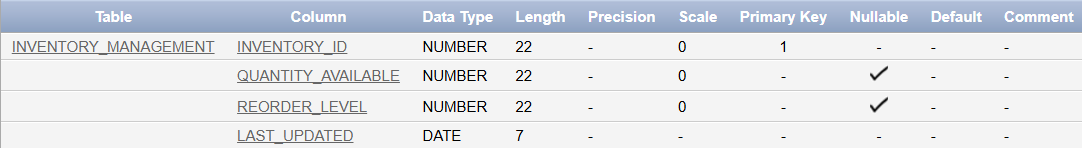
Reorder\_Level INT CHECK (Reorder\_Level >= 0),

Last\_Updated DATE NOT NULL

);

**Explanation:**

The Inventory\_Management table tracks medical supply stock. Inventory\_ID is the primary key. Attributes like Quantity\_Available and Reorder\_Level ensure efficient inventory monitoring. Constraints maintain logical values for stock levels.



**13. Medical\_Supply (Supply\_ID, Unit\_Cost, Quantity, Supply\_Name, Expiry\_Date, Inventory\_ID)**

**Code:**

CREATE TABLE Medical\_Supply (

Supply\_ID INT PRIMARY KEY,

Unit\_Cost NUMBER(10, 2) CHECK (Unit\_Cost >= 0),

Quantity INT CHECK (Quantity >= 0),

Supply\_Name VARCHAR2(100) NOT NULL,

Expiry\_Date DATE NOT NULL,

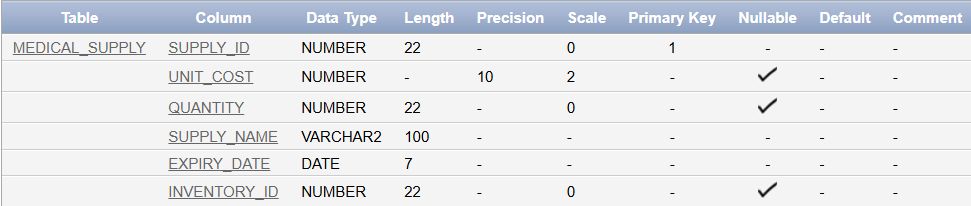
Inventory\_ID INT,

FOREIGN KEY (Inventory\_ID) REFERENCES Inventory\_Management(Inventory\_ID)

);

**Explanation:**

The Medical\_Supply table stores details of supplies. Supply\_ID is the primary key. Attributes such as Supply\_Name, Quantity, and Expiry\_Date track supply-specific details. A foreign key links supplies to inventory management (Inventory\_ID).



**14. Volunteer\_Schedule (Volunteer\_ID, Schedule\_ID, Start\_Time, End\_Time, Schedule\_Date)**

**Code:**

CREATE TABLE Volunteer\_Schedule (

Volunteer\_ID INT NOT NULL,

Schedule\_ID INT NOT NULL,

Start\_Time VARCHAR(4) NOT NULL,

End\_Time VARCHAR(4) NOT NULL,

Schedule\_Date DATE NOT NULL,

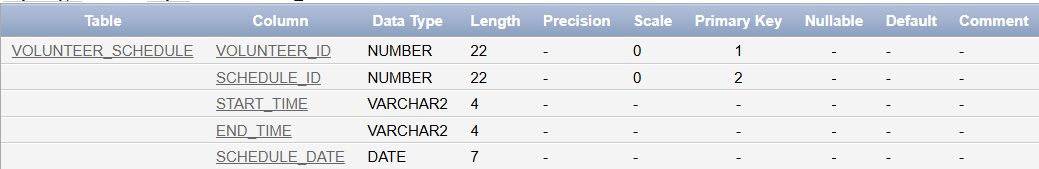
PRIMARY KEY (Volunteer\_ID, Schedule\_ID),

FOREIGN KEY (Volunteer\_ID) REFERENCES Volunteer(Volunteer\_ID)

);

**Explanation:**

The Volunteer\_Schedule table organizes volunteer work schedules. The composite primary key (Volunteer\_ID, Schedule\_ID) uniquely identifies each schedule. Attributes like Start\_Time, End\_Time, and Date capture scheduling details. A foreign key links schedules to volunteers.



**15. Volunteers\_For (Volunteer\_ID, Campaign\_ID, Participation\_Date, Role)**

**Code:**

CREATE TABLE Volunteers\_For (

Volunteer\_ID INT,

Campaign\_ID INT,

Participation\_Date DATE NOT NULL,

Role VARCHAR2(255),

PRIMARY KEY (Volunteer\_ID, Campaign\_ID),

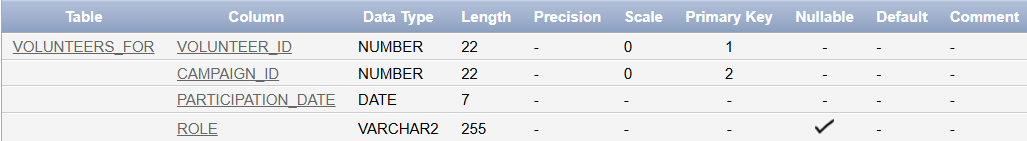
FOREIGN KEY (Volunteer\_ID) REFERENCES Volunteer(Volunteer\_ID),

FOREIGN KEY (Campaign\_ID) REFERENCES Campaign(Campaign\_ID)

);

**Explanation:**

The Volunteers\_For table tracks volunteer participation in campaigns. The composite primary key (Volunteer\_ID, Campaign\_ID) ensures unique records. Attributes like Participation\_Date and Role describe the involvement. Foreign keys ensure proper linkage to volunteers and campaigns.



**16. Partners\_With (Organization\_ID, Campaign\_ID, Start\_Date, Partnership\_Type)**

**Code:**

CREATE TABLE Partners\_With (

Organization\_ID INT,

Campaign\_ID INT,

Start\_Date DATE NOT NULL,

Partnership\_Type VARCHAR2(100) NOT NULL,

PRIMARY KEY (Organization\_ID, Campaign\_ID),

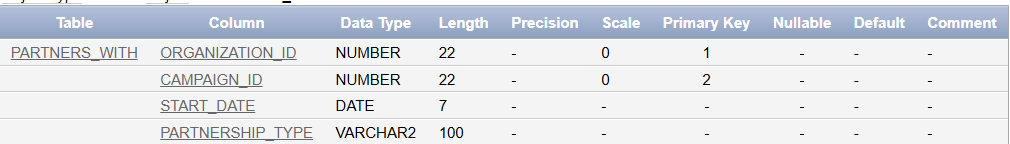
FOREIGN KEY (Organization\_ID) REFERENCES Organization (Organization\_ID),

FOREIGN KEY (Campaign\_ID) REFERENCES Campaign(Campaign\_ID)

);

**Explanation:**

The Partners\_With table documents partnerships between campaigns and organizations. The composite primary key (Organization\_ID, Campaign\_ID) uniquely identifies partnerships. Attributes like Start\_Date and Partnership\_Type describe the relationship.



**17. Requires (Supply\_ID, Event\_ID, Urgency, Status, Details, Supplies\_Requested)**

**Code:**

CREATE TABLE Requires (

Supply\_ID INT,

Event\_ID INT,

Urgency VARCHAR2(50) CHECK (Urgency IN ('Low', 'Medium', 'High')),

Status VARCHAR2(50),

Details VARCHAR2(255),

Supplies\_Requested VARCHAR2(255),

PRIMARY KEY (Supply\_ID, Event\_ID),

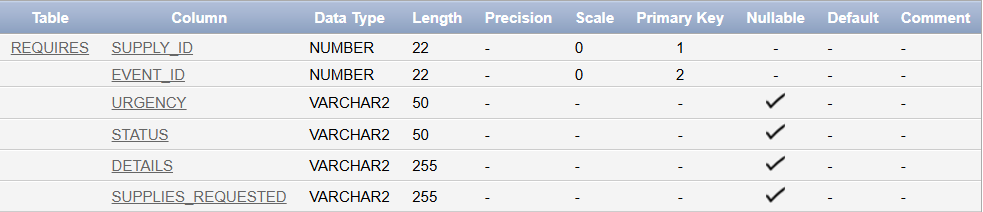
FOREIGN KEY (Supply\_ID) REFERENCES Medical\_Supply(Supply\_ID),

FOREIGN KEY (Event\_ID) REFERENCES Event(Event\_ID)

);

**Explanation:**

The Requires table tracks supplies needed for events. The composite primary key (Supply\_ID, Event\_ID) ensures unique records. Attributes like Urgency, Status, and Details provide context. Foreign keys link supplies and events.



**18. Requests (Request\_ID, Supply\_ID, Status, Details)**

**Code:**

CREATE TABLE Requests (

Request\_ID INT,

Supply\_ID INT,

Status VARCHAR2(50) NOT NULL,

Details VARCHAR2(255),

PRIMARY KEY (Request\_ID, Supply\_ID),

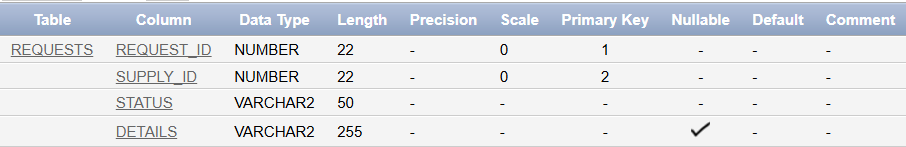
FOREIGN KEY (Request\_ID) REFERENCES Emergency\_Request(Request\_ID),

FOREIGN KEY (Supply\_ID) REFERENCES Medical\_Supply(Supply\_ID)

);

**Explanation:**

The Requests table manages supply requests related to emergencies. The composite primary key (Request\_ID, Supply\_ID) ensures unique entries. Attributes like Status and Details describe the request. Foreign keys link to the Emergency\_Request and Medical\_Supply tables.



**19. Assigned\_For (Patient\_ID, Volunteer\_ID, Role)**

**Code:**

CREATE TABLE Assigned\_For (

Patient\_ID INT,

Volunteer\_ID INT,

Role VARCHAR2(255) NOT NULL,

PRIMARY KEY (Patient\_ID, Volunteer\_ID),

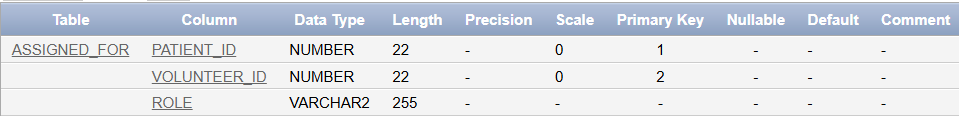
FOREIGN KEY (Patient\_ID) REFERENCES Patient(Patient\_ID),

FOREIGN KEY (Volunteer\_ID) REFERENCES Volunteer(Volunteer\_ID)

);

**Explanation:**

The Assigned\_For table tracks volunteer assignments to patients. The composite primary key (Patient\_ID, Volunteer\_ID) ensures unique entries. The Role attribute describes the assignment. Foreign keys link to the Patient and Volunteer tables.



**20. Participates\_In (Campaign\_ID, Patient\_ID)**

**Code:**

CREATE TABLE Participates\_In (

Campaign\_ID INT,

Patient\_ID INT,

PRIMARY KEY (Campaign\_ID, Patient\_ID),

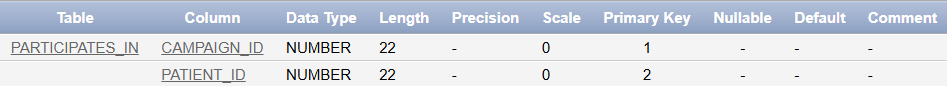
FOREIGN KEY (Campaign\_ID) REFERENCES Campaign(Campaign\_ID),

FOREIGN KEY (Patient\_ID) REFERENCES Patient(Patient\_ID)

);

**Explanation:**

The Participates\_In table tracks patient participation in campaigns. The composite primary key (Campaign\_ID, Patient\_ID) ensures unique entries. Foreign keys link patients to campaigns for accurate record-keeping.



**21. Participants (Training\_ID, Participant\_Name)**

**Code:**

CREATE TABLE Participants (

Training\_ID INT,

Participant\_Name VARCHAR2(100) NOT NULL,

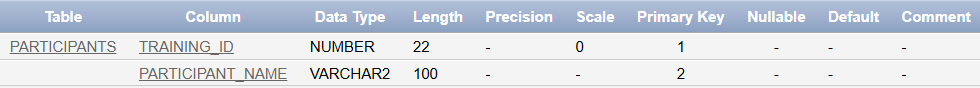
PRIMARY KEY (Training\_ID, Participant\_Name),

FOREIGN KEY (Training\_ID) REFERENCES Training\_Program(Training\_ID)

);

**Explanation:**

The Participants table documents attendees of training programs. The composite primary key (Training\_ID, Participant\_Name) ensures unique records. A foreign key links participants to the training programs they attended.



## Insertion of Data into the Database Tables

**Donor:**

INSERT INTO Donor VALUES (1, 'Ahmed', 'Hassan', 'M', 'ahmed.hassan@gmail.com', 'O+', '2024-01-01,2024-02-15');

INSERT INTO Donor VALUES (2, 'Leila', 'Khoury', 'A', 'leila.khoury@yahoo.com', 'A-', '2024-03-10');

INSERT INTO Donor VALUES (3, 'Omar', 'Nasser', 'K', 'omar.nasser@hotmail.com', 'B+', '2024-01-25,2024-02-20');

INSERT INTO Donor VALUES (4, 'Reem', 'Al-Masri', 'N', 'reem.masri@gmail.com', 'AB+', '2024-04-05');

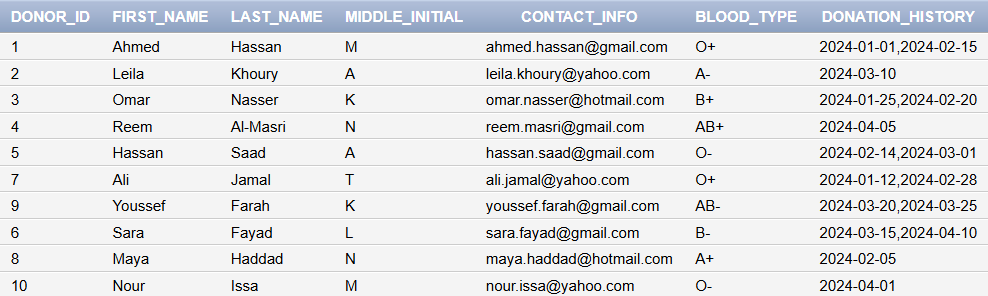
INSERT INTO Donor VALUES (5, 'Hassan', 'Saad', 'A', 'hassan.saad@gmail.com', 'O-', '2024-02-14,2024-03-01');

INSERT INTO Donor VALUES (6, 'Sara', 'Fayad', 'L', 'sara.fayad@gmail.com', 'B-', '2024-03-15,2024-04-10');

INSERT INTO Donor VALUES (7, 'Ali', 'Jamal', 'T', 'ali.jamal@yahoo.com', 'O+', '2024-01-12,2024-02-28');

INSERT INTO Donor VALUES (8, 'Maya', 'Haddad', 'N', 'maya.haddad@hotmail.com', 'A+', '2024-02-05');

INSERT INTO Donor VALUES (9, 'Youssef', 'Farah', 'K', 'youssef.farah@gmail.com', 'AB-', '2024-03-20,2024-03-25');

INSERT INTO Donor VALUES (10, 'Nour', 'Issa', 'M', 'nour.issa@yahoo.com', 'O-', '2024-04-01');

**Donation:**

INSERT INTO Donation VALUES (1, TO\_DATE('2024-01-01', 'YYYY-MM-DD'), 50.00, 'General Donation', 1, 1);

INSERT INTO Donation VALUES (2, TO\_DATE('2024-02-15', 'YYYY-MM-DD'), 75.00, 'Medical Supplies', 1, 2);

INSERT INTO Donation VALUES (3, TO\_DATE('2024-03-10', 'YYYY-MM-DD'), 100.00, 'Blood Drive Support', 2, 1);

INSERT INTO Donation VALUES (4, TO\_DATE('2024-01-25', 'YYYY-MM-DD'), 60.00, 'Emergency Fund', 3, 3);

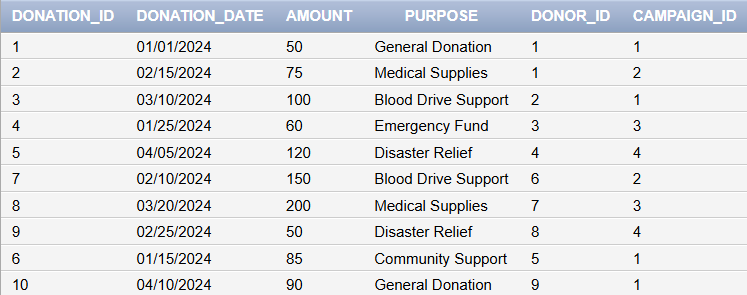
INSERT INTO Donation VALUES (5, TO\_DATE('2024-04-05', 'YYYY-MM-DD'), 120.00, 'Disaster Relief', 4, 4);

INSERT INTO Donation VALUES (6, TO\_DATE('2024-01-15', 'YYYY-MM-DD'), 85.00, 'Community Support', 5, 1);

INSERT INTO Donation VALUES (7, TO\_DATE('2024-02-10', 'YYYY-MM-DD'), 150.00, 'Blood Drive Support', 6, 2);

INSERT INTO Donation VALUES (8, TO\_DATE('2024-03-20', 'YYYY-MM-DD'), 200.00, 'Medical Supplies', 7, 3);

INSERT INTO Donation VALUES (9, TO\_DATE('2024-02-25', 'YYYY-MM-DD'), 50.00, 'Disaster Relief', 8, 4);

INSERT INTO Donation VALUES (10, TO\_DATE('2024-04-10', 'YYYY-MM-DD'), 90.00, 'General Donation', 9, 1);

**Volunteer:**

INSERT INTO Volunteer VALUES (1, 'Khaled', 'Al-Fayed', 'A', 'khaled.fayed@gmail.com', 'Weekends', 'Event Management', 1, 1);

INSERT INTO Volunteer VALUES (2, 'Amal', 'Haddad', 'M', 'amal.haddad@yahoo.com', 'Weekdays', 'Logistics', 2, 2);

INSERT INTO Volunteer VALUES (3, 'Zaid', 'Jaber', 'B', 'zaid.jaber@hotmail.com', 'Flexible', 'Blood Donation Drives', 3, 3);

INSERT INTO Volunteer VALUES (4, 'Nada', 'Al-Amin', 'L', 'nada.amin@gmail.com', 'Evenings', 'Medical Assistance', 4, 4);

INSERT INTO Volunteer VALUES (5, 'Sami', 'Rashid', 'C', 'sami.rashid@gmail.com', 'Mornings', 'Community Outreach', 5, 1);

INSERT INTO Volunteer VALUES (6, 'Hiba', 'Salim', 'E', 'hiba.salim@yahoo.com', 'Weekdays', 'Event Management', 6, 2);

INSERT INTO Volunteer VALUES (7, 'Rami', 'Kassem', 'F', 'rami.kassem@hotmail.com', 'Flexible', 'Logistics', 7, 3);

INSERT INTO Volunteer VALUES (8, 'Lara', 'Tamer', 'G', 'lara.tamer@gmail.com', 'Weekends', 'Medical Assistance', 8, 4);

INSERT INTO Volunteer VALUES (9, 'Jana', 'Moussa', 'H', 'jana.moussa@gmail.com', 'Evenings', 'Event Management', 9, 1);

INSERT INTO Volunteer VALUES (10, 'Fadi', 'Issam', 'I', 'fadi.issam@hotmail.com', 'Flexible', 'Blood Donation Drives', 10, 2);

**Patient:**

INSERT INTO Patient VALUES (1, 'Fatima', 'Khoury', 'M', 'Diabetes', 'O+', '078123456', 'fatima.khoury@gmail.com');

INSERT INTO Patient VALUES (2, 'Hadi', 'Jaber', 'A', 'Hypertension', 'A-', '070987654', 'hadi.jaber@gmail.com');

INSERT INTO Patient VALUES (3, 'Amira', 'Nasser', 'B', 'Asthma', 'B+', '071234567', 'amira.nasser@gmail.com');

INSERT INTO Patient VALUES (4, 'Ibrahim', 'Hassan', 'K', 'Anemia', 'AB+', '076543210', 'ibrahim.hassan@gmail.com');

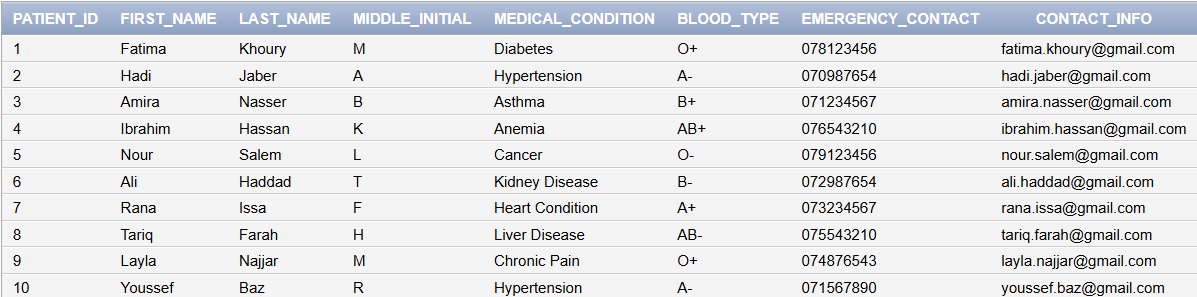
INSERT INTO Patient VALUES (5, 'Nour', 'Salem', 'L', 'Cancer', 'O-', '079123456', 'nour.salem@gmail.com');

INSERT INTO Patient VALUES (6, 'Ali', 'Haddad', 'T', 'Kidney Disease', 'B-', '072987654', 'ali.haddad@gmail.com');

INSERT INTO Patient VALUES (7, 'Rana', 'Issa', 'F', 'Heart Condition', 'A+', '073234567', 'rana.issa@gmail.com');

INSERT INTO Patient VALUES (8, 'Tariq', 'Farah', 'H', 'Liver Disease', 'AB-', '075543210', 'tariq.farah@gmail.com');

INSERT INTO Patient VALUES (9, 'Layla', 'Najjar', 'M', 'Chronic Pain', 'O+', '074876543', 'layla.najjar@gmail.com');

INSERT INTO Patient VALUES (10, 'Youssef', 'Baz', 'R', 'Hypertension', 'A-', '071567890', 'youssef.baz@gmail.com');

**Event:**

INSERT INTO Event VALUES (1, 'Annual Blood Donation', 'Beirut', 'Blood donation event held annually', TO\_DATE('2024-01-10', 'YYYY-MM-DD'), 1);

INSERT INTO Event VALUES (2, 'Relief Supplies Distribution', 'Tripoli', 'Distribution of medical relief supplies', TO\_DATE('2024-02-20', 'YYYY-MM-DD'), 2);

INSERT INTO Event VALUES (3, 'Disaster Preparedness Workshop', 'Saida', 'Training for disaster response', TO\_DATE('2024-02-15', 'YYYY-MM-DD'), 3);

INSERT INTO Event VALUES (4, 'Health Awareness Drive', 'Byblos', 'Promoting health awareness in the community', TO\_DATE('2024-03-25', 'YYYY-MM-DD'), 4);

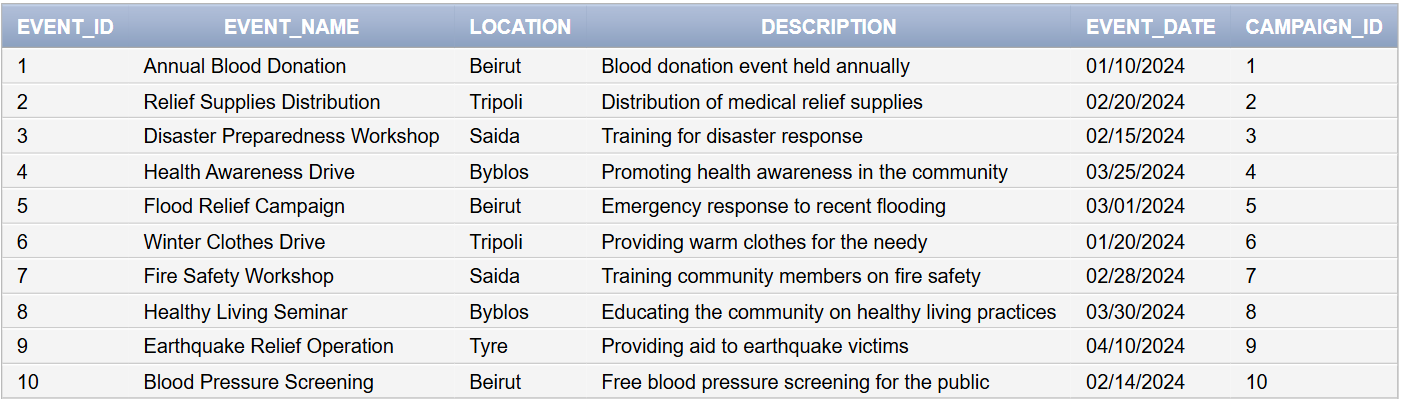
INSERT INTO Event VALUES (5, 'Flood Relief Campaign', 'Beirut', 'Emergency response to recent flooding', TO\_DATE('2024-03-01', 'YYYY-MM-DD'), 5);

INSERT INTO Event VALUES (6, 'Winter Clothes Drive', 'Tripoli', 'Providing warm clothes for the needy', TO\_DATE('2024-01-20', 'YYYY-MM-DD'), 6);

INSERT INTO Event VALUES (7, 'Fire Safety Workshop', 'Saida', 'Training community members on fire safety', TO\_DATE('2024-02-28', 'YYYY-MM-DD'), 7);

INSERT INTO Event VALUES (8, 'Healthy Living Seminar', 'Byblos', 'Educating the community on healthy living practices', TO\_DATE('2024-03-30', 'YYYY-MM-DD'), 8);

INSERT INTO Event VALUES (9, 'Earthquake Relief Operation', 'Tyre', 'Providing aid to earthquake victims', TO\_DATE('2024-04-10', 'YYYY-MM-DD'), 9);

INSERT INTO Event VALUES (10, 'Blood Pressure Screening', 'Beirut', 'Free blood pressure screening for the public', TO\_DATE('2024-02-14', 'YYYY-MM-DD'), 10);

**Emergency\_Request:**

INSERT INTO Emergency\_Request VALUES (1, TO\_DATE('2024-01-20', 'YYYY-MM-DD'), 'First Aid Kits, Bandages', 1);

INSERT INTO Emergency\_Request VALUES (2, TO\_DATE('2024-02-05', 'YYYY-MM-DD'), 'Blood Bags, Gloves', 2);

INSERT INTO Emergency\_Request VALUES (3, TO\_DATE('2024-02-18', 'YYYY-MM-DD'), 'Sanitizers, Masks', 3);

INSERT INTO Emergency\_Request VALUES (4, TO\_DATE('2024-03-10', 'YYYY-MM-DD'), 'Syringes, Medical Tape', 4);

INSERT INTO Emergency\_Request VALUES (5, TO\_DATE('2024-03-15', 'YYYY-MM-DD'), 'Water Bottles, Blankets', 5);

INSERT INTO Emergency\_Request VALUES (6, TO\_DATE('2024-03-20', 'YYYY-MM-DD'), 'Respirators, Oxygen Tanks', 6);

INSERT INTO Emergency\_Request VALUES (7, TO\_DATE('2024-03-25', 'YYYY-MM-DD'), 'Canned Food, Flashlights', 7);

INSERT INTO Emergency\_Request VALUES (8, TO\_DATE('2024-04-01', 'YYYY-MM-DD'), 'IV Drips, Glucose Packets', 8);

INSERT INTO Emergency\_Request VALUES (9, TO\_DATE('2024-04-05', 'YYYY-MM-DD'), 'Defibrillators, ECG Machines', 9);

INSERT INTO Emergency\_Request VALUES (10, TO\_DATE('2024-04-10', 'YYYY-MM-DD'), 'Sleeping Bags, Tents', 10);

**Training\_Program:**

INSERT INTO Training\_Program VALUES (1, 'Training on blood donation management', 'Blood Donation Basics', TO\_DATE('2024-01-05', 'YYYY-MM-DD'), 4);

INSERT INTO Training\_Program VALUES (2, 'Workshop on organizing disaster events', 'Disaster Management', TO\_DATE('2024-01-12', 'YYYY-MM-DD'), 6);

INSERT INTO Training\_Program VALUES (3, 'Logistics training for volunteers', 'Logistics and Coordination', TO\_DATE('2024-02-01', 'YYYY-MM-DD'), 3);

INSERT INTO Training\_Program VALUES (4, 'Medical training for handling emergencies', 'Emergency Medical Assistance', TO\_DATE('2024-03-01', 'YYYY-MM-DD'), 5);

INSERT INTO Training\_Program VALUES (5, 'Community outreach training', 'Effective Communication', TO\_DATE('2024-02-10', 'YYYY-MM-DD'), 2);

INSERT INTO Training\_Program VALUES (6, 'Workshop on first aid basics', 'First Aid Essentials', TO\_DATE('2024-02-20', 'YYYY-MM-DD'), 4);

INSERT INTO Training\_Program VALUES (7, 'Training on mental health support', 'Mental Health First Aid', TO\_DATE('2024-03-10', 'YYYY-MM-DD'), 3);

INSERT INTO Training\_Program VALUES (8, 'Advanced logistics and supply chain management', 'Advanced Logistics', TO\_DATE('2024-03-15', 'YYYY-MM-DD'), 6);

INSERT INTO Training\_Program VALUES (9, 'Disaster relief and recovery training', 'Disaster Recovery', TO\_DATE('2024-04-01', 'YYYY-MM-DD'), 5);

INSERT INTO Training\_Program VALUES (10, 'Emergency evacuation planning workshop', 'Evacuation Planning', TO\_DATE('2024-04-10', 'YYYY-MM-DD'), 7);

**Disaster\_Event:**

INSERT INTO Disaster\_Event VALUES (1, 'Flood response in Beirut', TO\_DATE('2024-02-10', 'YYYY-MM-DD'), 'High', 'Beirut', 'Flood Relief');

INSERT INTO Disaster\_Event VALUES (2, 'Fire response in Tripoli', TO\_DATE('2024-02-25', 'YYYY-MM-DD'), 'Medium', 'Tripoli', 'Fire Relief');

INSERT INTO Disaster\_Event VALUES (3, 'Earthquake response in Saida', TO\_DATE('2024-03-05', 'YYYY-MM-DD'), 'High', 'Saida', 'Earthquake Relief');

INSERT INTO Disaster\_Event VALUES (4, 'Pandemic awareness drive in Byblos', TO\_DATE('2024-03-20', 'YYYY-MM-DD'), 'Low', 'Byblos', 'Pandemic Awareness');

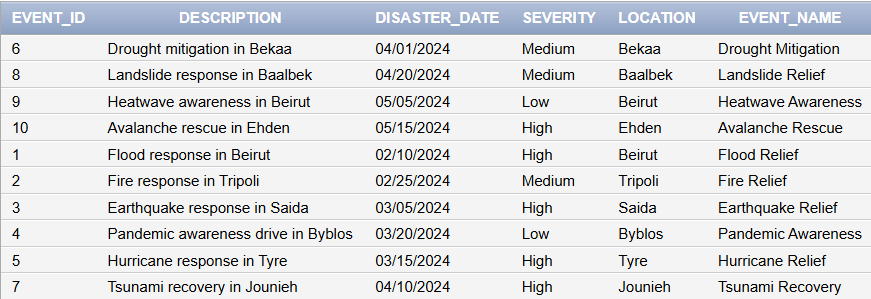
INSERT INTO Disaster\_Event VALUES (5, 'Hurricane response in Tyre', TO\_DATE('2024-03-15', 'YYYY-MM-DD'), 'High', 'Tyre', 'Hurricane Relief');

INSERT INTO Disaster\_Event VALUES (6, 'Drought mitigation in Bekaa', TO\_DATE('2024-04-01', 'YYYY-MM-DD'), 'Medium', 'Bekaa', 'Drought Mitigation');

INSERT INTO Disaster\_Event VALUES (7, 'Tsunami recovery in Jounieh', TO\_DATE('2024-04-10', 'YYYY-MM-DD'), 'High', 'Jounieh', 'Tsunami Recovery');

INSERT INTO Disaster\_Event VALUES (8, 'Landslide response in Baalbek', TO\_DATE('2024-04-20', 'YYYY-MM-DD'), 'Medium', 'Baalbek', 'Landslide Relief');

INSERT INTO Disaster\_Event VALUES (9, 'Heatwave awareness in Beirut', TO\_DATE('2024-05-05', 'YYYY-MM-DD'), 'Low', 'Beirut', 'Heatwave Awareness');

INSERT INTO Disaster\_Event VALUES (10, 'Avalanche rescue in Ehden', TO\_DATE('2024-05-15', 'YYYY-MM-DD'), 'High', 'Ehden', 'Avalanche Rescue');

**Feedback:**

INSERT INTO Feedback VALUES (1, 'Very well organized!', 5, TO\_DATE('2024-01-15', 'YYYY-MM-DD'), 1, 1);

INSERT INTO Feedback VALUES (2, 'Good support for the community.', 4, TO\_DATE('2024-02-28', 'YYYY-MM-DD'), 2, 2);

INSERT INTO Feedback VALUES (3, 'Helpful training session.', 5, TO\_DATE('2024-03-01', 'YYYY-MM-DD'), 3, 3);

INSERT INTO Feedback VALUES (4, 'Great event, well-executed.', 5, TO\_DATE('2024-03-25', 'YYYY-MM-DD'), 4);

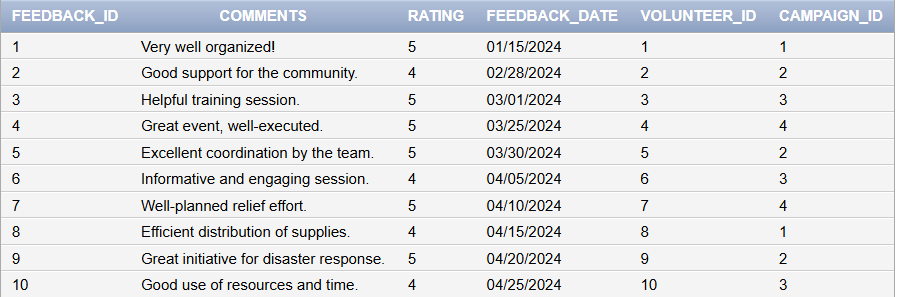
INSERT INTO Feedback VALUES (5, 'Excellent coordination by the team.', 5, TO\_DATE('2024-03-30', 'YYYY-MM-DD'), 5, 2);

INSERT INTO Feedback VALUES (6, 'Informative and engaging session.', 4, TO\_DATE('2024-04-05', 'YYYY-MM-DD'), 6, 3);

INSERT INTO Feedback VALUES (7, 'Well-planned relief effort.', 5, TO\_DATE('2024-04-10', 'YYYY-MM-DD'), 7, 4);

INSERT INTO Feedback VALUES (8, 'Efficient distribution of supplies.', 4, TO\_DATE('2024-04-15', 'YYYY-MM-DD'), 8, 1);

INSERT INTO Feedback VALUES (9, 'Great initiative for disaster response.', 5, TO\_DATE('2024-04-20', 'YYYY-MM-DD'), 9, 2);

INSERT INTO Feedback VALUES (10, 'Good use of resources and time.', 4, TO\_DATE('2024-04-25', 'YYYY-MM-DD'), 10, 3);

**Campaign:**

INSERT INTO Campaign VALUES (1, 'Blood Drive 2024', TO\_DATE('2024-01-01', 'YYYY-MM-DD'), TO\_DATE('2024-03-01', 'YYYY-MM-DD'), 'Annual blood donation campaign');

INSERT INTO Campaign VALUES (2, 'Medical Relief Fund', TO\_DATE('2024-02-01', 'YYYY-MM-DD'), TO\_DATE('2024-04-01', 'YYYY-MM-DD'), 'Raising funds for medical supplies');

INSERT INTO Campaign VALUES (3, 'Emergency Response', TO\_DATE('2024-01-15', 'YYYY-MM-DD'), TO\_DATE('2024-03-15', 'YYYY-MM-DD'), 'Emergency disaster response support');

INSERT INTO Campaign VALUES (4, 'Community Health Outreach', TO\_DATE('2024-03-01', 'YYYY-MM-DD'), TO\_DATE('2024-05-01', 'YYYY-MM-DD'), 'Promoting community health initiatives');

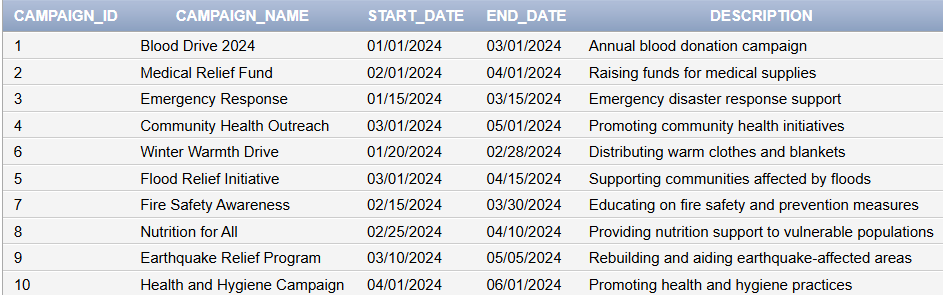
INSERT INTO Campaign VALUES (5, 'Flood Relief Initiative', TO\_DATE('2024-03-01', 'YYYY-MM-DD'), TO\_DATE('2024-04-15', 'YYYY-MM-DD'), 'Supporting communities affected by floods');

INSERT INTO Campaign VALUES (6, 'Winter Warmth Drive', TO\_DATE('2024-01-20', 'YYYY-MM-DD'), TO\_DATE('2024-02-28', 'YYYY-MM-DD'), 'Distributing warm clothes and blankets');

INSERT INTO Campaign VALUES (7, 'Fire Safety Awareness', TO\_DATE('2024-02-15', 'YYYY-MM-DD'), TO\_DATE('2024-03-30', 'YYYY-MM-DD'), 'Educating on fire safety and prevention measures');

INSERT INTO Campaign VALUES (8, 'Nutrition for All', TO\_DATE('2024-02-25', 'YYYY-MM-DD'), TO\_DATE('2024-04-10', 'YYYY-MM-DD'), 'Providing nutrition support to vulnerable populations');

INSERT INTO Campaign VALUES (9, 'Earthquake Relief Program', TO\_DATE('2024-03-10', 'YYYY-MM-DD'), TO\_DATE('2024-05-05', 'YYYY-MM-DD'), 'Rebuilding and aiding earthquake-affected areas');

INSERT INTO Campaign VALUES (10, 'Health and Hygiene Campaign', TO\_DATE('2024-04-01', 'YYYY-MM-DD'), TO\_DATE('2024-06-01', 'YYYY-MM-DD'), 'Promoting health and hygiene practices');

**Organization:**

INSERT INTO Organization VALUES (1, 'Red Crescent', 'info@redcrescent.org');

INSERT INTO Organization VALUES (2, 'Beirut Health Foundation', 'contact@bhf.org');

INSERT INTO Organization VALUES (3, 'Saida Disaster Relief', 'support@saida.org');

INSERT INTO Organization VALUES (4, 'Byblos Wellness Initiative', 'info@bybloswellness.com');

INSERT INTO Organization VALUES (5, 'Lebanon Relief Fund', 'contact@lebanonrelief.org');

INSERT INTO Organization VALUES (6, 'Tripoli Aid Society', 'info@tripoliaid.org');

INSERT INTO Organization VALUES (7, 'Tyre Medical Support', 'support@tyremedical.org');

INSERT INTO Organization VALUES (8, 'Baalbek Disaster Aid', 'info@baalbekaid.org');

INSERT INTO Organization VALUES (9, 'Mount Lebanon Health', 'contact@mountlebanonhealth.org');

INSERT INTO Organization VALUES (10, 'Bekaa Valley Outreach', 'support@bekaaoutreach.org');

**Inventory\_Management:**

INSERT INTO Inventory\_Management VALUES (1, 500, 100, TO\_DATE('2024-01-10', 'YYYY-MM-DD'));

INSERT INTO Inventory\_Management VALUES (2, 300, 50, TO\_DATE('2024-02-15', 'YYYY-MM-DD'));

INSERT INTO Inventory\_Management VALUES (3, 700, 200, TO\_DATE('2024-02-20', 'YYYY-MM-DD'));

INSERT INTO Inventory\_Management VALUES (4, 450, 100, TO\_DATE('2024-03-01', 'YYYY-MM-DD'));

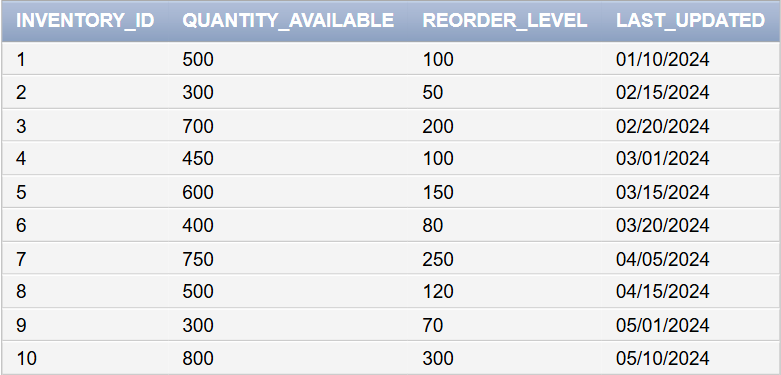
INSERT INTO Inventory\_Management VALUES (5, 600, 150, TO\_DATE('2024-03-15', 'YYYY-MM-DD'));

INSERT INTO Inventory\_Management VALUES (6, 400, 80, TO\_DATE('2024-03-20', 'YYYY-MM-DD'));

INSERT INTO Inventory\_Management VALUES (7, 750, 250, TO\_DATE('2024-04-05', 'YYYY-MM-DD'));

INSERT INTO Inventory\_Management VALUES (8, 500, 120, TO\_DATE('2024-04-15', 'YYYY-MM-DD'));

INSERT INTO Inventory\_Management VALUES (9, 300, 70, TO\_DATE('2024-05-01', 'YYYY-MM-DD'));

INSERT INTO Inventory\_Management VALUES (10, 800, 300, TO\_DATE('2024-05-10', 'YYYY-MM-DD'));

**Medical\_Supply:**

INSERT INTO Medical\_Supply VALUES (1, 2.50, 200, 'Bandages', TO\_DATE('2025-01-01', 'YYYY-MM-DD'), 1);

INSERT INTO Medical\_Supply VALUES (2, 5.00, 100, 'Blood Bags', TO\_DATE('2025-02-01', 'YYYY-MM-DD'), 2);

INSERT INTO Medical\_Supply VALUES (3, 1.50, 300, 'Gloves', TO\_DATE('2024-12-01', 'YYYY-MM-DD'), 3);

INSERT INTO Medical\_Supply VALUES (4, 3.00, 150, 'Sanitizers', TO\_DATE('2025-03-01', 'YYYY-MM-DD'), 4);

INSERT INTO Medical\_Supply VALUES (5, 4.00, 250, 'Face Masks', TO\_DATE('2025-06-01', 'YYYY-MM-DD'), 5);

INSERT INTO Medical\_Supply VALUES (6, 6.50, 120, 'Syringes', TO\_DATE('2025-04-15', 'YYYY-MM-DD'), 6);

INSERT INTO Medical\_Supply VALUES (7, 8.00, 80, 'IV Drips', TO\_DATE('2025-05-10', 'YYYY-MM-DD'), 7);

INSERT INTO Medical\_Supply VALUES (8, 10.00, 60, 'Defibrillators', TO\_DATE('2025-07-01', 'YYYY-MM-DD'), 8);

INSERT INTO Medical\_Supply VALUES (9, 2.00, 300, 'Medical Tape', TO\_DATE('2024-12-20', 'YYYY-MM-DD'), 9);

INSERT INTO Medical\_Supply VALUES (10, 7.50, 100, 'Oxygen Tanks', TO\_DATE('2025-03-30', 'YYYY-MM-DD'), 10);

**Volunteers\_Schedule:**

INSERT INTO Volunteer\_Schedule VALUES (1, 101, TO\_DATE('08:00:00', 'HH24:MI:SS'), TO\_DATE('12:00:00', 'HH24:MI:SS'), TO\_DATE('2024-01-10', 'YYYY-MM-DD'));

INSERT INTO Volunteer\_Schedule VALUES (2, 102, TO\_DATE('09:00:00', 'HH24:MI:SS'), TO\_DATE('13:00:00', 'HH24:MI:SS'), TO\_DATE('2024-01-15', 'YYYY-MM-DD'));

INSERT INTO Volunteer\_Schedule VALUES (3, 103, TO\_DATE('10:00:00', 'HH24:MI:SS'), TO\_DATE('14:00:00', 'HH24:MI:SS'), TO\_DATE('2024-01-20', 'YYYY-MM-DD'));

INSERT INTO Volunteer\_Schedule VALUES (4, 104, TO\_DATE('11:00:00', 'HH24:MI:SS'), TO\_DATE('15:00:00', 'HH24:MI:SS'), TO\_DATE('2024-01-25', 'YYYY-MM-DD'));

INSERT INTO Volunteer\_Schedule VALUES (5, 105, TO\_DATE('08:30:00', 'HH24:MI:SS'), TO\_DATE('12:30:00', 'HH24:MI:SS'), TO\_DATE('2024-02-01', 'YYYY-MM-DD'));

INSERT INTO Volunteer\_Schedule VALUES (6, 106, TO\_DATE('09:30:00', 'HH24:MI:SS'), TO\_DATE('13:30:00', 'HH24:MI:SS'), TO\_DATE('2024-02-05', 'YYYY-MM-DD'));

INSERT INTO Volunteer\_Schedule VALUES (7, 107, TO\_DATE('07:00:00', 'HH24:MI:SS'), TO\_DATE('11:00:00', 'HH24:MI:SS'), TO\_DATE('2024-02-10', 'YYYY-MM-DD'));

INSERT INTO Volunteer\_Schedule VALUES (8, 108, TO\_DATE('12:00:00', 'HH24:MI:SS'), TO\_DATE('16:00:00', 'HH24:MI:SS'), TO\_DATE('2024-02-15', 'YYYY-MM-DD'));

INSERT INTO Volunteer\_Schedule VALUES (9, 109, TO\_DATE('13:00:00', 'HH24:MI:SS'), TO\_DATE('17:00:00', 'HH24:MI:SS'), TO\_DATE('2024-02-20', 'YYYY-MM-DD'));

INSERT INTO Volunteer\_Schedule VALUES (10, 110, TO\_DATE('14:00:00', 'HH24:MI:SS'), TO\_DATE('18:00:00', 'HH24:MI:SS'), TO\_DATE('2024-02-25', 'YYYY-MM-DD'));

**Volunteers\_For:**

INSERT INTO Volunteers\_For VALUES (1, 1, TO\_DATE('2024-01-15', 'YYYY-MM-DD'), 'Organizer');

INSERT INTO Volunteers\_For VALUES (2, 2, TO\_DATE('2024-02-20', 'YYYY-MM-DD'), 'Distributor');

INSERT INTO Volunteers\_For VALUES (3, 3, TO\_DATE('2024-02-25', 'YYYY-MM-DD'), 'Trainer');

INSERT INTO Volunteers\_For VALUES (4, 4, TO\_DATE('2024-03-05', 'YYYY-MM-DD'), 'Medical Assistant');

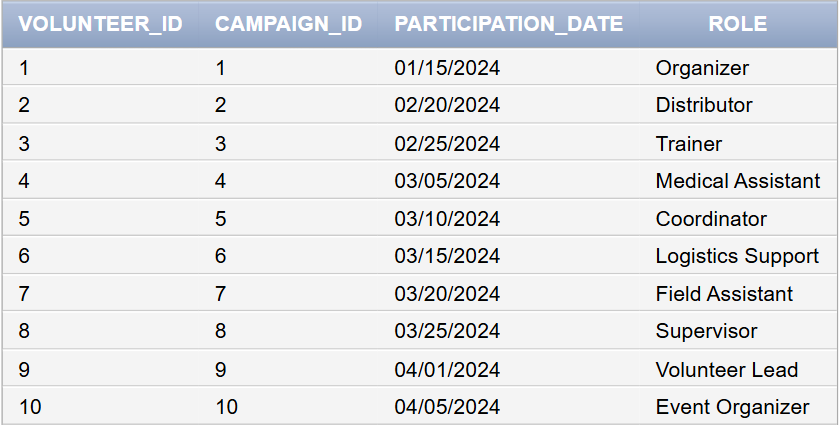
INSERT INTO Volunteers\_For VALUES (5, 5, TO\_DATE('2024-03-10', 'YYYY-MM-DD'), 'Coordinator');

INSERT INTO Volunteers\_For VALUES (6, 6, TO\_DATE('2024-03-15', 'YYYY-MM-DD'), 'Logistics Support');

INSERT INTO Volunteers\_For VALUES (7, 7, TO\_DATE('2024-03-20', 'YYYY-MM-DD'), 'Field Assistant');

INSERT INTO Volunteers\_For VALUES (8, 8, TO\_DATE('2024-03-25', 'YYYY-MM-DD'), 'Supervisor');

INSERT INTO Volunteers\_For VALUES (9, 9, TO\_DATE('2024-04-01', 'YYYY-MM-DD'), 'Volunteer Lead');

INSERT INTO Volunteers\_For VALUES (10, 10, TO\_DATE('2024-04-05', 'YYYY-MM-DD'), 'Event Organizer');****

**Partners\_With:**

INSERT INTO Partners\_With VALUES (1, 1, TO\_DATE('2024-01-01', 'YYYY-MM-DD'), 'Financial Support');

INSERT INTO Partners\_With VALUES (2, 2, TO\_DATE('2024-01-15', 'YYYY-MM-DD'), 'Logistics Assistance');

INSERT INTO Partners\_With VALUES (3, 3, TO\_DATE('2024-02-01', 'YYYY-MM-DD'), 'Awareness Campaign');

INSERT INTO Partners\_With VALUES (4, 4, TO\_DATE('2024-03-01', 'YYYY-MM-DD'), 'Medical Supplies');

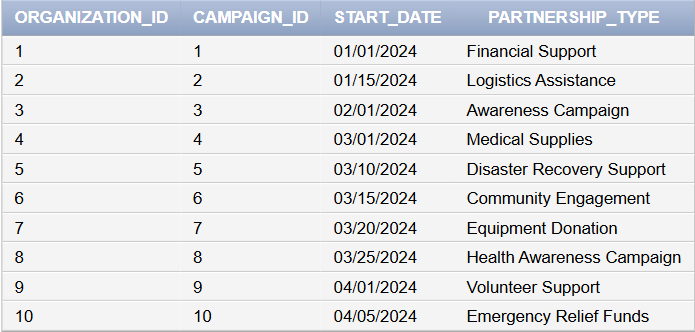
INSERT INTO Partners\_With VALUES (5, 5, TO\_DATE('2024-03-10', 'YYYY-MM-DD'), 'Disaster Recovery Support');

INSERT INTO Partners\_With VALUES (6, 6, TO\_DATE('2024-03-15', 'YYYY-MM-DD'), 'Community Engagement');

INSERT INTO Partners\_With VALUES (7, 7, TO\_DATE('2024-03-20', 'YYYY-MM-DD'), 'Equipment Donation');

INSERT INTO Partners\_With VALUES (8, 8, TO\_DATE('2024-03-25', 'YYYY-MM-DD'), 'Health Awareness Campaign');

INSERT INTO Partners\_With VALUES (9, 9, TO\_DATE('2024-04-01', 'YYYY-MM-DD'), 'Volunteer Support');

INSERT INTO Partners\_With VALUES (10, 10, TO\_DATE('2024-04-05', 'YYYY-MM-DD'), 'Emergency Relief Funds');

**Requires:**

INSERT INTO Requires VALUES (1, 1, 'High', 'Pending', 'For disaster relief', 'Bandages, Gloves');

INSERT INTO Requires VALUES (2, 2, 'Medium', 'Delivered', 'For blood donation', 'Blood Bags');

INSERT INTO Requires VALUES (3, 3, 'High', 'Pending', 'For earthquake victims', 'First Aid Kits');

INSERT INTO Requires VALUES (4, 4, 'Low', 'Delivered', 'For pandemic awareness', 'Sanitizers, Masks');

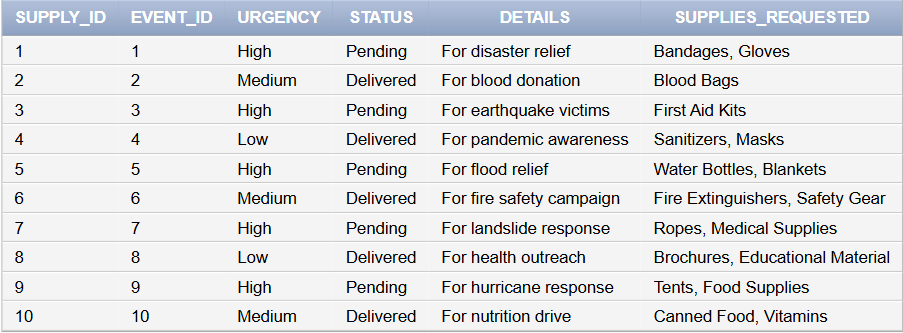
INSERT INTO Requires VALUES (5, 5, 'High', 'Pending', 'For flood relief', 'Water Bottles, Blankets');

INSERT INTO Requires VALUES (6, 6, 'Medium', 'Delivered', 'For fire safety campaign', 'Fire Extinguishers, Safety Gear');

INSERT INTO Requires VALUES (7, 7, 'High', 'Pending', 'For landslide response', 'Ropes, Medical Supplies');

INSERT INTO Requires VALUES (8, 8, 'Low', 'Delivered', 'For health outreach', 'Brochures, Educational Material');

INSERT INTO Requires VALUES (9, 9, 'High', 'Pending', 'For hurricane response', 'Tents, Food Supplies');

INSERT INTO Requires VALUES (10, 10, 'Medium', 'Delivered', 'For nutrition drive', 'Canned Food, Vitamins');

**Requests:**

INSERT INTO Requests VALUES (1, 1, 'Pending', 'Needed for flood relief');

INSERT INTO Requests VALUES (2, 2, 'Delivered', 'For blood donation drive');

INSERT INTO Requests VALUES (3, 3, 'Pending', 'Required for earthquake victims');

INSERT INTO Requests VALUES (4, 4, 'Delivered', 'For pandemic relief supplies');

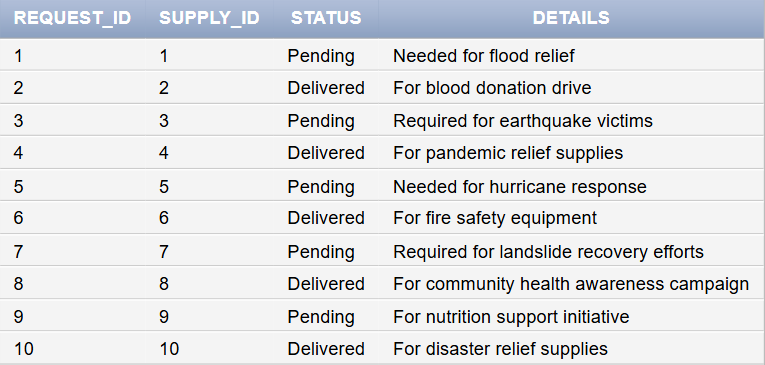
INSERT INTO Requests VALUES (5, 5, 'Pending', 'Needed for hurricane response');

INSERT INTO Requests VALUES (6, 6, 'Delivered', 'For fire safety equipment');

INSERT INTO Requests VALUES (7, 7, 'Pending', 'Required for landslide recovery efforts');

INSERT INTO Requests VALUES (8, 8, 'Delivered', 'For community health awareness campaign');

INSERT INTO Requests VALUES (9, 9, 'Pending', 'For nutrition support initiative');

INSERT INTO Requests VALUES (10, 10, 'Delivered', 'For disaster relief supplies');

**Assigned For:**

INSERT INTO Assigned\_For VALUES (1, 1, 'Caregiver');

INSERT INTO Assigned\_For VALUES (2, 2, 'Logistics Support');

INSERT INTO Assigned\_For VALUES (3, 3, 'Medical Assistant');

INSERT INTO Assigned\_For VALUES (4, 4, 'Trainer');

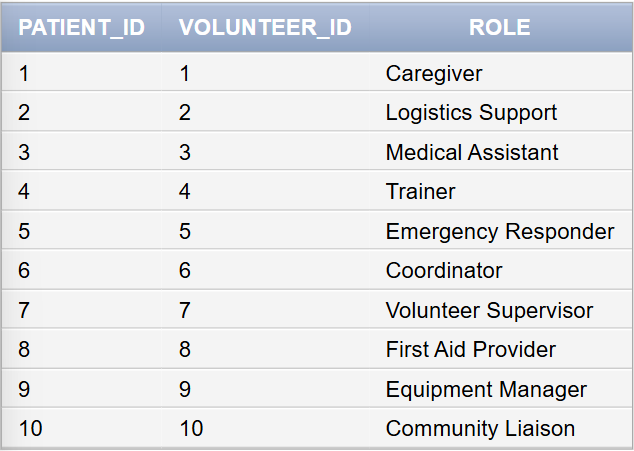
INSERT INTO Assigned\_For VALUES (5, 5, 'Emergency Responder');

INSERT INTO Assigned\_For VALUES (6, 6, 'Coordinator');

INSERT INTO Assigned\_For VALUES (7, 7, 'Volunteer Supervisor');

INSERT INTO Assigned\_For VALUES (8, 8, 'First Aid Provider');

INSERT INTO Assigned\_For VALUES (9, 9, 'Equipment Manager');

INSERT INTO Assigned\_For VALUES (10, 10, 'Community Liaison');

**Participates in:**

INSERT INTO Participates\_In VALUES (1, 1);

INSERT INTO Participates\_In VALUES (2, 2);

INSERT INTO Participates\_In VALUES (3, 3);

INSERT INTO Participates\_In VALUES (4, 4);

INSERT INTO Participates\_In VALUES (5, 5);

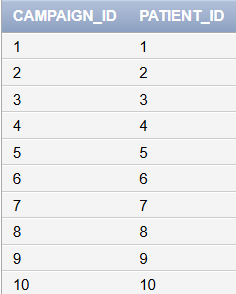
INSERT INTO Participates\_In VALUES (6, 6);

INSERT INTO Participates\_In VALUES (7, 7);

INSERT INTO Participates\_In VALUES (8, 8);

INSERT INTO Participates\_In VALUES (9, 9);

INSERT INTO Participates\_In VALUES (10, 10);



**Participants:**

INSERT INTO Participants VALUES (1, 'Khaled Al-Fayed');

INSERT INTO Participants VALUES (2, 'Amal Haddad');

INSERT INTO Participants VALUES (3, 'Zaid Jaber');

INSERT INTO Participants VALUES (4, 'Nada Al-Amin');

INSERT INTO Participants VALUES (5, 'Sami Rashid');

INSERT INTO Participants VALUES (6, 'Hiba Salim');

INSERT INTO Participants VALUES (7, 'Rami Kassem');

INSERT INTO Participants VALUES (8, 'Lara Tamer');

INSERT INTO Participants VALUES (9, 'Jana Moussa');

INSERT INTO Participants VALUES (10, 'Fadi Issam');

## Queries

**1. Rewarding the Most Active Volunteers**

To motivate and acknowledge the dedication of volunteers, the organization has decided to reward the top contributors who participated in the highest number of campaigns. By identifying the volunteers with the most campaign involvements, the company can ensure fair recognition of their efforts.

SELECT v.Volunteer\_ID, v.First\_Name, v.Last\_Name, COUNT(DISTINCT vf.Campaign\_ID) AS Campaign\_Count

FROM Volunteers\_For vf

JOIN Volunteer v ON vf.Volunteer\_ID = v.Volunteer\_ID

GROUP BY v.Volunteer\_ID, v.First\_Name, v.Last\_Name

ORDER BY Campaign\_Count DESC;

**2. Total Medical Supplies Used in Events Organized by Specific Campaigns**

In Tripoli, concerns have been voiced about the mismanagement of medical supplies during the "Blood Drive 2024" campaign. To investigate, the company must examine the total quantity of supplies utilized in all events organized under this campaign to determine whether excessive quantities were misallocated or wasted.

SELECT SUM(Quantity) AS Total\_Supplies

FROM Medical\_Supply

WHERE Supply\_ID IN (

SELECT Supply\_ID

FROM Requires

WHERE Event\_ID IN (

SELECT Event\_ID

FROM Event

WHERE Campaign\_ID = (

SELECT Campaign\_ID

FROM Campaign

WHERE Campaign\_Name = 'Blood Drive 2024'

)

)

);



**3. Volunteers Assigned to Patients Diagnosed with Specific Medical Conditions**

A recent incident in Byblos highlighted potential mismatches between volunteers' expertise and the medical conditions of the patients they were assigned to. For example, diabetes patients might have been paired with volunteers lacking the appropriate training. To resolve this, a detailed list of volunteers, their assignments, and the relevant patient conditions must be reviewed.

SELECT v.Volunteer\_ID, v.First\_Name, v.Last\_Name, p.Patient\_ID, p.Medical\_Condition

FROM Volunteer v

JOIN Assigned\_For af ON v.Volunteer\_ID = af.Volunteer\_ID

JOIN Patient p ON af.Patient\_ID = p.Patient\_ID

WHERE p.Medical\_Condition = 'Diabetes'; 

**4. Campaigns with Events Requiring Urgent Medical Supplies**

A shortage of critical medical supplies during a high-severity flood relief event in Saida has drawn public criticism. To mitigate future risks, the campaigns that organized such events requiring urgent supplies must be thoroughly investigated to ensure proper preparation for emergencies.

SELECT DISTINCT Campaign\_Name

FROM Campaign

WHERE Campaign\_ID IN (

SELECT Campaign\_ID

FROM Event

WHERE Event\_ID IN (

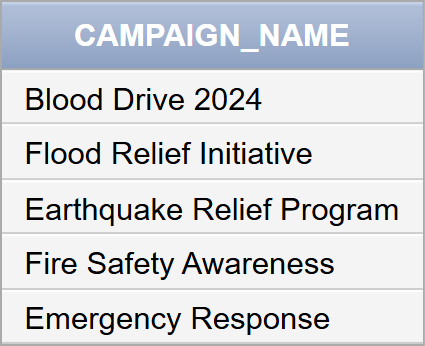
SELECT Event\_ID

FROM Requires

WHERE Urgency = 'High'

)

);



**5. Patients Participating in Campaigns They Are Assigned To**

Several patients in Beirut claimed they were asked to participate in campaigns unrelated to their medical conditions. To validate these claims, the company must investigate whether patients were indeed participating in the same campaigns they were assigned to or if mismatches occurred.

SELECT Patient\_ID, First\_Name, Last\_Name

FROM Patient

WHERE Patient\_ID IN (

SELECT Patient\_ID

FROM Participates\_In

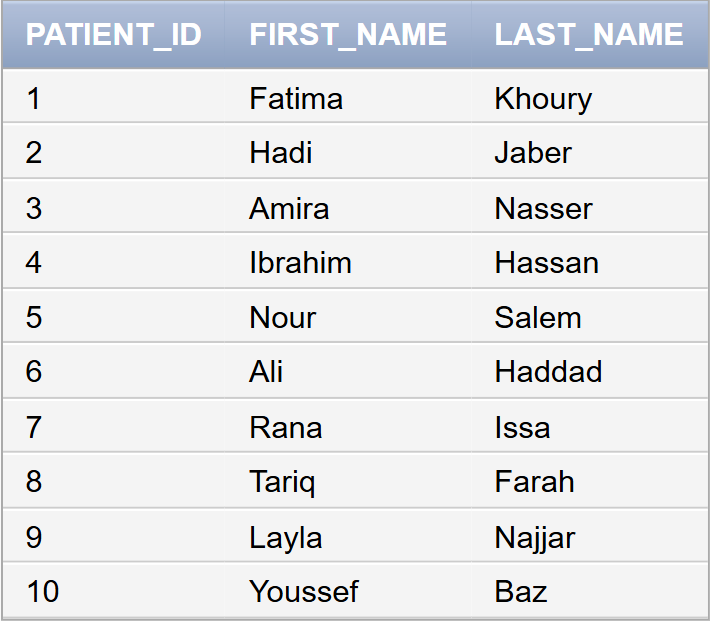
WHERE Campaign\_ID IN (

SELECT Campaign\_ID

FROM Assigned\_For

WHERE Patient\_ID = Participates\_In.Patient\_ID

)

);

**6. Volunteers Scheduled for Events on the Same Date**

Complaints from volunteers in Byblos indicate scheduling conflicts where volunteers were scheduled for events on overlapping or incorrect dates. The company must investigate whether any volunteers had schedules overlapping with events they were assigned to.

SELECT vs.Volunteer\_ID, vs.Schedule\_ID, vs.Schedule\_Date

FROM Volunteer\_Schedule vs

WHERE vs.Schedule\_Date IN (

SELECT e.Event\_Date

FROM Event e

WHERE e.Event\_ID IN (

SELECT v.Disaster\_Event\_ID

FROM Volunteer v

WHERE v.Volunteer\_ID = vs.Volunteer\_ID

    )

);

**7. Supplies with Low Stock Requested for Disaster Events**

A review in Beirut uncovered issues with low-stock supplies being repeatedly requested for disaster events, such as gloves and bandages during earthquake relief. To address this issue, supplies with insufficient inventory levels must be analyzed for better resource allocation.

SELECT Supply\_Name

FROM Medical\_Supply

WHERE Supply\_ID IN (

SELECT Supply\_ID

FROM Requires

WHERE Event\_ID IN (

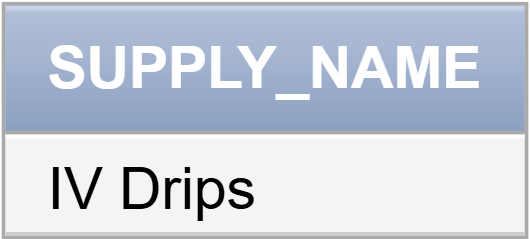
SELECT Event\_ID

FROM Disaster\_Event

WHERE Severity = 'High'

)

) AND Quantity < 100;



**8. Organizations Supporting Campaigns with High-Rated Feedback**

Organizations such as "Red Crescent" and "Beirut Health Foundation" have claimed significant contributions to highly-rated campaigns. To validate these claims, the company must identify campaigns with excellent feedback and the organizations supporting them.

SELECT Organization\_Name

FROM Organization

WHERE Organization\_ID IN (

SELECT Organization\_ID

FROM Partners\_With

WHERE Campaign\_ID IN (

SELECT Campaign\_ID

FROM Feedback

WHERE Rating >= 4

)

);

**9. Volunteers Who Assisted in the Largest Number of Disaster Events**

After the Beirut Port Explosion in 2020, many volunteers also helped during other crises in Lebanon, like the wildfires in Chouf or flood relief in Tripoli. Identifying those who participated in the most disaster events helps recognize their efforts and assign them important roles in future emergencies. This ensures their experience is used effectively in disaster response.

SELECT v.Volunteer\_ID, v.First\_Name, v.Last\_Name, COUNT(de.Event\_ID) AS Disaster\_Count

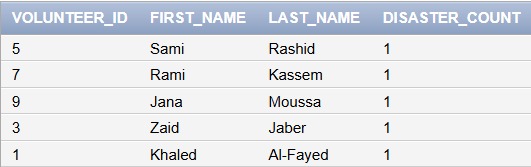
FROM Volunteer v

JOIN Disaster\_Event de ON v.Disaster\_Event\_ID = de.Event\_ID

WHERE de.Severity = 'High'

GROUP BY v.Volunteer\_ID, v.First\_Name, v.Last\_Name

ORDER BY Disaster\_Count DESC;



**10. Volunteers Providing Feedback for Campaigns They Helped Organize**

In Tripoli, questions have arisen about whether volunteers who provided feedback for campaigns they organized were unbiased. The company must identify these volunteers to ensure transparency and credibility in campaign reviews.

SELECT Volunteer\_ID, First\_Name, Last\_Name

FROM Volunteer

WHERE Volunteer\_ID IN (

SELECT Volunteer\_ID

FROM Feedback

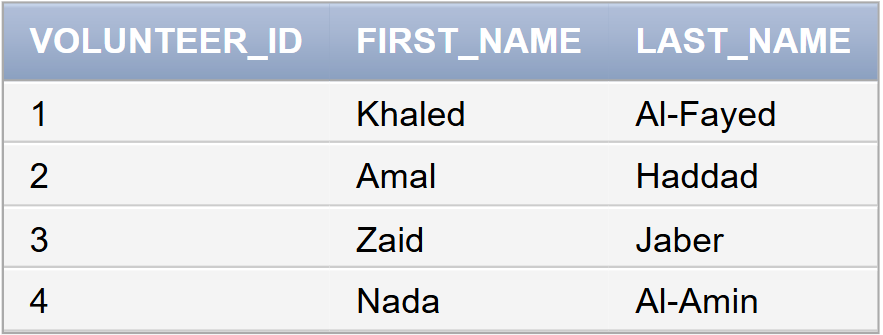
WHERE Campaign\_ID IN (

SELECT Campaign\_ID

FROM Volunteers\_For

WHERE Volunteer\_ID = Feedback.Volunteer\_ID

)

);

# Phase 4: Normalization for Relational Databases

After creating all the relations, they should be normalized up to the four levels of normalization, Boyce- Codd Normal Form, to enhance them.

Data normalization involves organizing data in a database to minimize redundancy and dependency by organizing fields and tables of a database. This helps in avoiding anomalies, such as update anomalies, insertion anomalies, and deletion anomalies, which can occur when data is not properly normalized. Thus, data normalization also indirectly contributes to reducing page I/O time.

First Normal Form:

Multivalued attributes, composite attributes, and their combinations cannot exist in a relationship with this form.

1. Single atomic values are the only accepted attribute values.

2. An attribute's domain can only include atomic values, and an attribute's value in a tuple can only contain one value from that domain.

3. Prohibits using a collection of values for a single tuple's attribute value.

Second Normal Form:

The Second Normal Form is determined by the concept of complete functional dependence. To achieve this level, two conditions must be satisfied: first every non-prime attribute in R must be totally functionally dependent on each key in R, and second, every non-prime attribute A in R should not be partially dependent on any key in R. The terms functional dependency, prime attribute, full functional dependency, and partial dependency are defined as follows:

Functional Dependencies: A restriction involving two sets of database attributes. The values of an X component influence or determine the values of the Y component of a tuple in relation to R. We state that Y depends on X functionally.

Prime attribute: An attribute of a relation R that is a part of a primary key or involved in a candidate key. When an attribute is not a prime, that is when it is not a part of any primary 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.

Third Normal Form:

A relation is said to be in the third normal form if it is already in the second normal form and if no non-prime attribute (an attribute not part of any candidate key) is transitively dependent on the primary key. A transitive dependency occurs when one non-prime attribute is functionally dependent on another non-prime attribute, rather than directly on the primary key. If P -> Q and Q -> R is true, then P-> R is a transitive dependency. For every non-trivial functional dependency X →Y either X should be a super key or Y is a prime attribute.

Boyce- Codd Normal Form:

In comparison to the third normal form, the Boyce-Codd normal form is more stringent. Such that the third normal form allows the right-hand side of the functional dependency to be a prime attribute, BCNF prohibits this.

### **1. Donor Table**

**Original Table**:  
**Donor (Donor\_ID, Name, Blood\_Type, Contact\_Info, Donation\_History)**

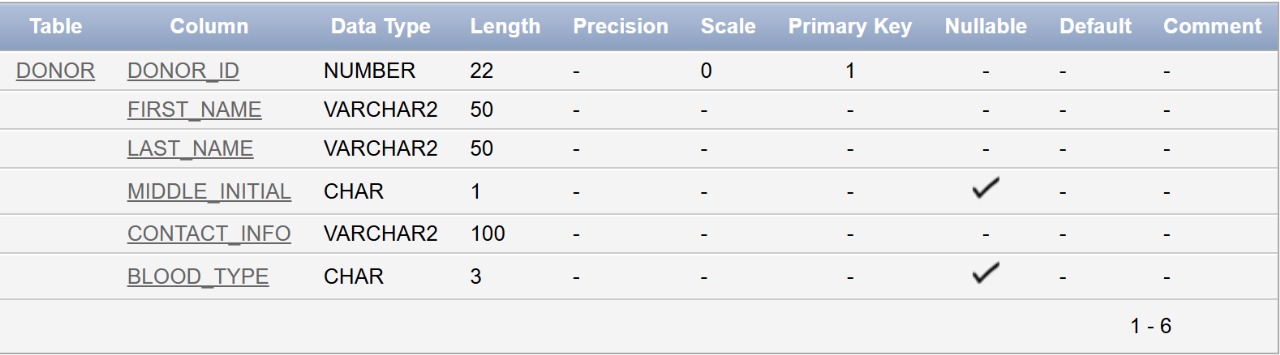
#### **Check for 1NF**:

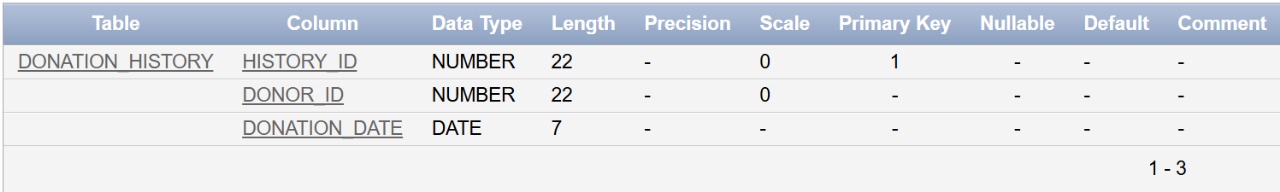
* The attribute **Donation\_History** contains multiple values (e.g., a list of donation dates), which violates atomicity.
* **Does not satisfy 1NF**.

#### **Normalization for 1NF**:

* Split **Donation\_History** into a new table where each row represents a single donation date.

**Normalized Tables**:





#### **Check for 2NF**:

* After splitting **Donation\_History**, all non-key attributes in both tables fully depend on their respective primary keys.
* **Satisfies 2NF**.

#### **Check for 3NF**:

* No transitive dependencies exist.
* **Satisfies 3NF**.

#### **Check for BCNF**:

* All functional dependencies have superkeys as determinants.
* **Satisfies BCNF**.

### **2. Campaign Table**

**Original Table**:  
**Campaign (Campaign\_ID, Campaign\_Name, Start\_Date, End\_Date, Description)**

#### **Check for 1NF**:

* The table contains atomic values and no multi-valued attributes or repeating groups.
* **Satisfies 1NF**.

#### **Check for 2NF**:

* The primary key is **Campaign\_ID**, and all non-key attributes fully depend on it.
* **Satisfies 2NF**.

#### **Check for 3NF**:

* No transitive dependencies exist.
* **Satisfies 3NF**.

#### **Check for BCNF**:

* All functional dependencies have superkeys as determinants.
* **Satisfies BCNF**.

### **3. Donation Table**

**Original Table**:  
**Donation (Donation\_ID, Donor\_ID, Campaign\_ID, Donation\_Date, Amount, Purpose)**

#### **Check for 1NF**:

* The table contains atomic values and no multi-valued attributes or repeating groups.
* **Satisfies 1NF**.

#### **Check for 2NF**:

* The primary key is **Donation\_ID**, and all non-key attributes fully depend on it.
* **Satisfies 2NF**.

#### **Check for 3NF**:

* No transitive dependencies exist.
* **Satisfies 3NF**.

#### **Check for BCNF**:

* All functional dependencies have superkeys as determinants.
* **Satisfies BCNF**.

### **4. Volunteer Table**

**Original Table**:  
**Volunteer (Volunteer\_ID, Name, Contact\_Info, Availability, Volunteer\_Interest, Training\_ID, Disaster\_Event\_ID)**

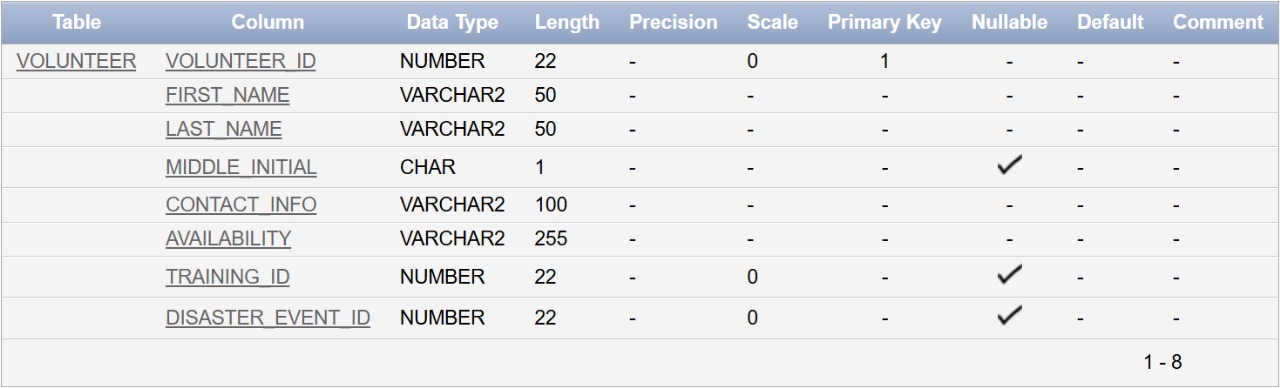
#### **Check for 1NF**:

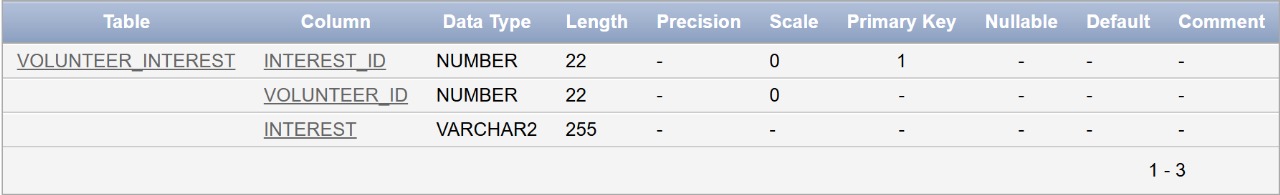
* **Volunteer\_Interest** might store multiple values (e.g., a list of interests), which violates atomicity.
* **Does not satisfy 1NF**.

#### **Normalization for 1NF**:

* Split **Volunteer\_Interest** into a new table with one row per interest.

**Normalized Tables**:





#### **Check for 2NF**:

* All non-key attributes in both tables fully depend on their respective primary keys.
* **Satisfies 2NF**.

#### **Check for 3NF**:

* No transitive dependencies exist.
* **Satisfies 3NF**.

#### **Check for BCNF**:

* All functional dependencies have superkeys as determinants.
* **Satisfies BCNF**.

### **5. Patient Table**

**Original Table**:  
**Patient (Patient\_ID, Name, Medical\_Condition, Blood\_Type, Emergency\_Contact, Contact\_Info)**

#### **Check for 1NF**:

* The table contains atomic values and no multi-valued attributes or repeating groups.
* **Satisfies 1NF**.

#### **Check for 2NF**:

* The primary key is **Patient\_ID**, and all non-key attributes fully depend on it.
* **Satisfies 2NF**.

#### **Check for 3NF**:

* No transitive dependencies exist.
* **Satisfies 3NF**.

#### **Check for BCNF**:

* All functional dependencies have superkeys as determinants.
* **Satisfies BCNF**.

### **6. Event Table**

**Original Table**:  
**Event (Event\_ID, Event\_Name, Location, Description, Event\_Date, Campaign\_ID)**

#### **Check for 1NF**:

* The table contains atomic values and no multi-valued attributes or repeating groups.
* **Satisfies 1NF**.

#### **Check for 2NF**:

* The primary key is **Event\_ID**, and all non-key attributes fully depend on it.
* **Satisfies 2NF**.

#### **Check for 3NF**:

* No transitive dependencies exist.
* **Satisfies 3NF**.

#### **Check for BCNF**:

* All functional dependencies have superkeys as determinants.
* **Satisfies BCNF**.

### **7. Emergency\_Request Table**

**Original Table**:  
**Emergency\_Request (Request\_ID, Request\_Date, Requested\_Supplies, Requesting\_Volunteer\_ID)**

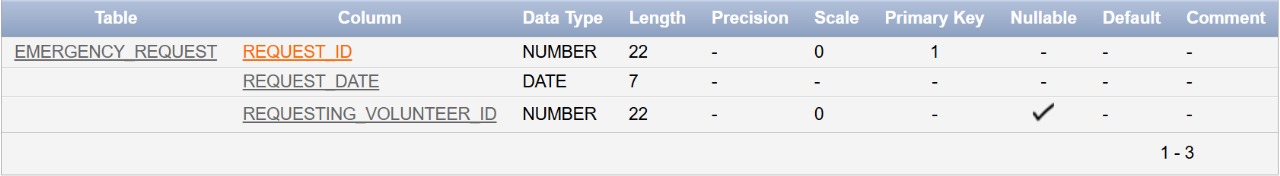
#### **Check for 1NF**:

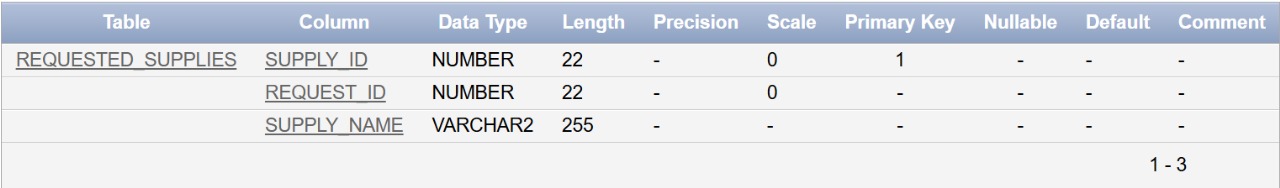
* **Requested\_Supplies** may contain multiple values (e.g., a list of supplies), which violates atomicity.
* **Does not satisfy 1NF**.

#### **Normalization for 1NF**:

* Move **Requested\_Supplies** into a new table where each row represents a single supply.

**Normalized Tables**:





#### **Check for 2NF**:

* All non-key attributes in both tables fully depend on their respective primary keys.
* **Satisfies 2NF**.

#### **Check for 3NF**:

* No transitive dependencies exist.
* **Satisfies 3NF**.

#### **Check for BCNF**:

* All functional dependencies have superkeys as determinants.
* **Satisfies BCNF**.

### **8. Training\_Program Table**

**Original Table**:  
**Training\_Program (Training\_ID, Description, Title, Training\_Date, Duration)**

#### **Check for 1NF**:

* The table contains atomic values and no multi-valued attributes or repeating groups.
* **Satisfies 1NF**.

#### **Check for 2NF**:

* The primary key is **Training\_ID**, and all non-key attributes fully depend on it.
* **Satisfies 2NF**.

#### **Check for 3NF**:

* No transitive dependencies exist.
* **Satisfies 3NF**.

#### **Check for BCNF**:

* All functional dependencies have superkeys as determinants.
* **Satisfies BCNF**.

### **9. Disaster\_Event Table**

**Original Table**:  
**Disaster\_Event (Event\_ID, Description, Severity, Location, Event\_Name)**

#### **Check for 1NF**:

* The table contains atomic values and no multi-valued attributes or repeating groups.
* **Satisfies 1NF**.

#### **Check for 2NF**:

* The primary key is **Event\_ID**, and all non-key attributes fully depend on it.
* **Satisfies 2NF**.

#### **Check for 3NF**:

* No transitive dependencies exist.
* **Satisfies 3NF**.

#### **Check for BCNF**:

* All functional dependencies have superkeys as determinants.
* **Satisfies BCNF**.

### **10. Feedback Table**

**Original Table**:  
**Feedback (Feedback\_ID, Comments, Rating, Feedback\_Date, Volunteer\_ID, Campaign\_ID)**

#### **Check for 1NF**:

* The table contains atomic values and no multi-valued attributes or repeating groups.
* **Satisfies 1NF**.

#### **Check for 2NF**:

* The primary key is **Feedback\_ID**, and all non-key attributes fully depend on it.
* **Satisfies 2NF**.

#### **Check for 3NF**:

* No transitive dependencies exist.
* **Satisfies 3NF**.

#### **Check for BCNF**:

* All functional dependencies have superkeys as determinants.
* **Satisfies BCNF**.

### **11. Organization Table**

**Original Table**:  
**Organization (Organization\_ID, Organization\_Name, Contact\_Info)**

#### **Check for 1NF**:

* The table contains atomic values and no multi-valued attributes or repeating groups.
* **Satisfies 1NF**.

#### **Check for 2NF**:

* The primary key is **Organization\_ID**, and all non-key attributes fully depend on it.
* **Satisfies 2NF**.

#### **Check for 3NF**:

* No transitive dependencies exist.
* **Satisfies 3NF**.

#### **Check for BCNF**:

* All functional dependencies have superkeys as determinants.
* **Satisfies BCNF**.

### **12. Inventory\_Management Table**

**Original Table**:  
**Inventory\_Management (Inventory\_ID, Quantity\_Available, Reorder\_Level, Last\_Updated)**

#### **Check for 1NF**:

* The table contains atomic values and no multi-valued attributes or repeating groups.
* **Satisfies 1NF**.

#### **Check for 2NF**:

* The primary key is **Inventory\_ID**, and all non-key attributes fully depend on it.
* **Satisfies 2NF**.

#### **Check for 3NF**:

* No transitive dependencies exist.
* **Satisfies 3NF**.

#### **Check for BCNF**:

* All functional dependencies have superkeys as determinants.
* **Satisfies BCNF**.

### **13. Medical\_Supply Table**

**Original Table**:  
**Medical\_Supply (Supply\_ID, Unit\_Cost, Quantity, Supply\_Name, Expiry\_Date, Inventory\_ID)**

#### **Check for 1NF**:

* The table contains atomic values and no multi-valued attributes or repeating groups.
* **Satisfies 1NF**.

#### **Check for 2NF**:

* The primary key is **Supply\_ID**, and all non-key attributes fully depend on it.
* **Satisfies 2NF**.

#### **Check for 3NF**:

* No transitive dependencies exist.
* **Satisfies 3NF**.

#### **Check for BCNF**:

* All functional dependencies have superkeys as determinants.
* **Satisfies BCNF**.

### **14. Volunteer\_Schedule Table**

**Original Table**:  
**Volunteer\_Schedule (Volunteer\_ID, Schedule\_ID, Start\_Time, End\_Time, Schedule\_Date)**

#### **Check for 1NF**:

* The table contains atomic values and no multi-valued attributes or repeating groups.
* **Satisfies 1NF**.

#### **Check for 2NF**:

* The composite primary key is **(Volunteer\_ID, Schedule\_ID)**, and all non-key attributes fully depend on it.
* **Satisfies 2NF**.

#### **Check for 3NF**:

* No transitive dependencies exist.
* **Satisfies 3NF**.

#### **Check for BCNF**:

* All functional dependencies have superkeys as determinants.
* **Satisfies BCNF**.

### **15. Volunteers\_For Table**

**Original Table**:  
**Volunteers\_For (Volunteer\_ID, Campaign\_ID, Participation\_Date, Role)**

#### **Check for 1NF**:

* The table contains atomic values and no multi-valued attributes or repeating groups.
* **Satisfies 1NF**.

#### **Check for 2NF**:

* The composite primary key is **(Volunteer\_ID, Campaign\_ID)**, and all non-key attributes fully depend on it.
* **Satisfies 2NF**.

#### **Check for 3NF**:

* No transitive dependencies exist.
* **Satisfies 3NF**.

#### **Check for BCNF**:

* All functional dependencies have superkeys as determinants.
* **Satisfies BCNF**.

### **16. Partners\_With Table**

**Original Table**:  
**Partners\_With (Organization\_ID, Campaign\_ID, Start\_Date, Partnership\_Type)**

#### **Check for 1NF**:

* The table contains atomic values and no multi-valued attributes or repeating groups.
* **Satisfies 1NF**.

#### **Check for 2NF**:

* The composite primary key is **(Organization\_ID, Campaign\_ID)**, and all non-key attributes fully depend on it.
* **Satisfies 2NF**.

#### **Check for 3NF**:

* No transitive dependencies exist.
* **Satisfies 3NF**.

#### **Check for BCNF**:

* All functional dependencies have superkeys as determinants.
* **Satisfies BCNF**.

### **17. Requires Table**

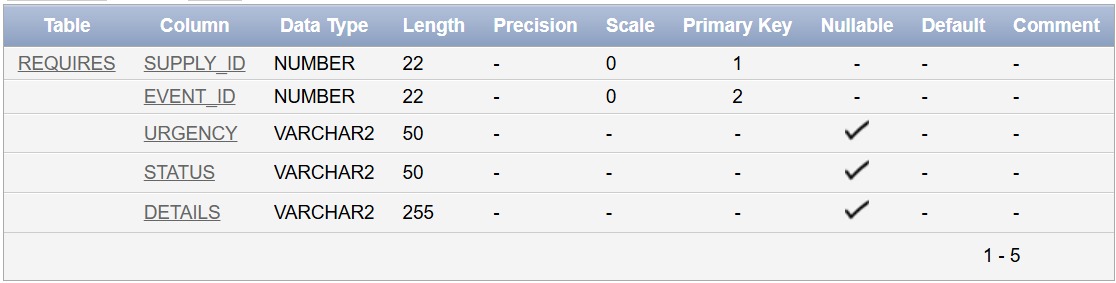
**Original Table**:  
**Requires (Supply\_ID, Event\_ID, Urgency, Status, Details, Supplies\_Requested)**

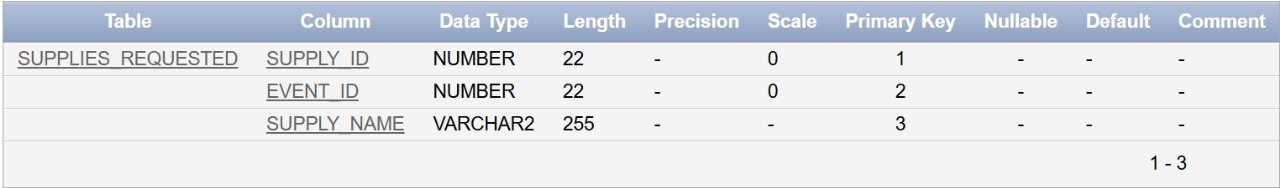
#### **Check for 1NF**:

* The attribute **Supplies\_Requested** likely contains multiple values (e.g., a list of supplies), violating the atomicity rule.
* **Does not satisfy 1NF**.

#### **Normalization for 1NF**:

* Split **Supplies\_Requested** into a new table where each row represents one requested supply.

**Normalized Tables**: 



#### **Check for 2NF**:

* After normalization, all non-key attributes in both tables depend fully on their respective primary keys.
* **Satisfies 2NF**.

#### **Check for 3NF**:

* No transitive dependencies exist.
* **Satisfies 3NF**.

#### **Check for BCNF**:

* All functional dependencies have superkeys as determinants.
* **Satisfies BCNF**.

### **18. Requests Table**

**Original Table**:  
**Requests (Request\_ID, Supply\_ID, Status, Details)**

#### **Check for 1NF**:

* The table contains atomic values and no multi-valued attributes or repeating groups.
* **Satisfies 1NF**.

#### **Check for 2NF**:

* The composite primary key is **(Request\_ID, Supply\_ID)**, and all non-key attributes depend fully on this key.
* **Satisfies 2NF**.

#### **Check for 3NF**:

* No transitive dependencies exist.
* **Satisfies 3NF**.

#### **Check for BCNF**:

* All functional dependencies have superkeys as determinants.
* **Satisfies BCNF**.

### **19. Assigned\_For Table**

**Original Table**:  
**Assigned\_For (Patient\_ID, Volunteer\_ID, Role)**

#### **Check for 1NF**:

* The table contains atomic values and no multi-valued attributes or repeating groups.
* **Satisfies 1NF**.

#### **Check for 2NF**:

* The composite primary key is **(Patient\_ID, Volunteer\_ID)**, and all non-key attributes depend fully on this key.
* **Satisfies 2NF**.

#### **Check for 3NF**:

* No transitive dependencies exist.
* **Satisfies 3NF**.

#### **Check for BCNF**:

* All functional dependencies have superkeys as determinants.
* **Satisfies BCNF**.

### **20. Participates\_In Table**

**Original Table**:  
**Participates\_In (Campaign\_ID, Patient\_ID)**

#### **Check for 1NF**:

* The table contains atomic values and no multi-valued attributes or repeating groups.
* **Satisfies 1NF**.

#### **Check for 2NF**:

* The composite primary key is **(Campaign\_ID, Patient\_ID)**, and there are no partial dependencies.
* **Satisfies 2NF**.

#### **Check for 3NF**:

* No transitive dependencies exist.
* **Satisfies 3NF**.

#### **Check for BCNF**:

* All functional dependencies have superkeys as determinants.
* **Satisfies BCNF**.

### **21. Participants Table**

**Original Table**:  
**Participants (Training\_ID, Participant\_Name)**

#### **Check for 1NF**:

* The table contains atomic values and no multi-valued attributes or repeating groups.
* **Satisfies 1NF**.

#### **Check for 2NF**:

* The composite primary key is **(Training\_ID, Participant\_Name)**, and there are no partial dependencies.
* **Satisfies 2NF**.

#### **Check for 3NF**:

* No transitive dependencies exist.
* **Satisfies 3NF**.

#### **Check for BCNF**:

* All functional dependencies have superkeys as determinants.
* **Satisfies BCNF**.

**Summary of Changes:**

For all 21 tables, we checked compliance with **1NF**, **2NF**, **3NF**, and **BCNF**. The following tables required normalization:

1. **Donor**: Split **Donation\_History** into a new table.
2. **Volunteer**: Split **Volunteer\_Interest** into a new table.
3. **Emergency\_Request**: Split **Supplies\_Requested** into a new table.
4. **Requires**: Split **Supplies\_Requested** into a new table.