Disaster Response Coordination Database Project Documentation

Project done by member of group number 1

1. Project Idea Definition

1.1. Detailed Explanation

Problem: In recent years, Kenya has faced recurring disasters such as floods, droughts, and landslides, often resulting in significant humanitarian and infrastructural challenges.

The lack of a centralized system to coordinate response efforts, allocate resources, and manage data on affected individuals and organizations often hampers effective disaster response.

Our project aims to develop a relational database system to support Disaster Response Coordination. The system will centralize data related to disaster events, response teams, aid distribution, and affected individuals. This supports SDG 11 (Sustainable Cities and Communities) and SDG 13 (Climate Action) by improving disaster preparedness and community resilience.

1.2. Scope and Objectives

Scope:
The system will be used by government agencies, NGOs, local authorities, and emergency responders to manage and track disasters, relief resources (food, medicine), disaster response activities (e.g. relocation) across various affected regions of Kenya.

Objectives:

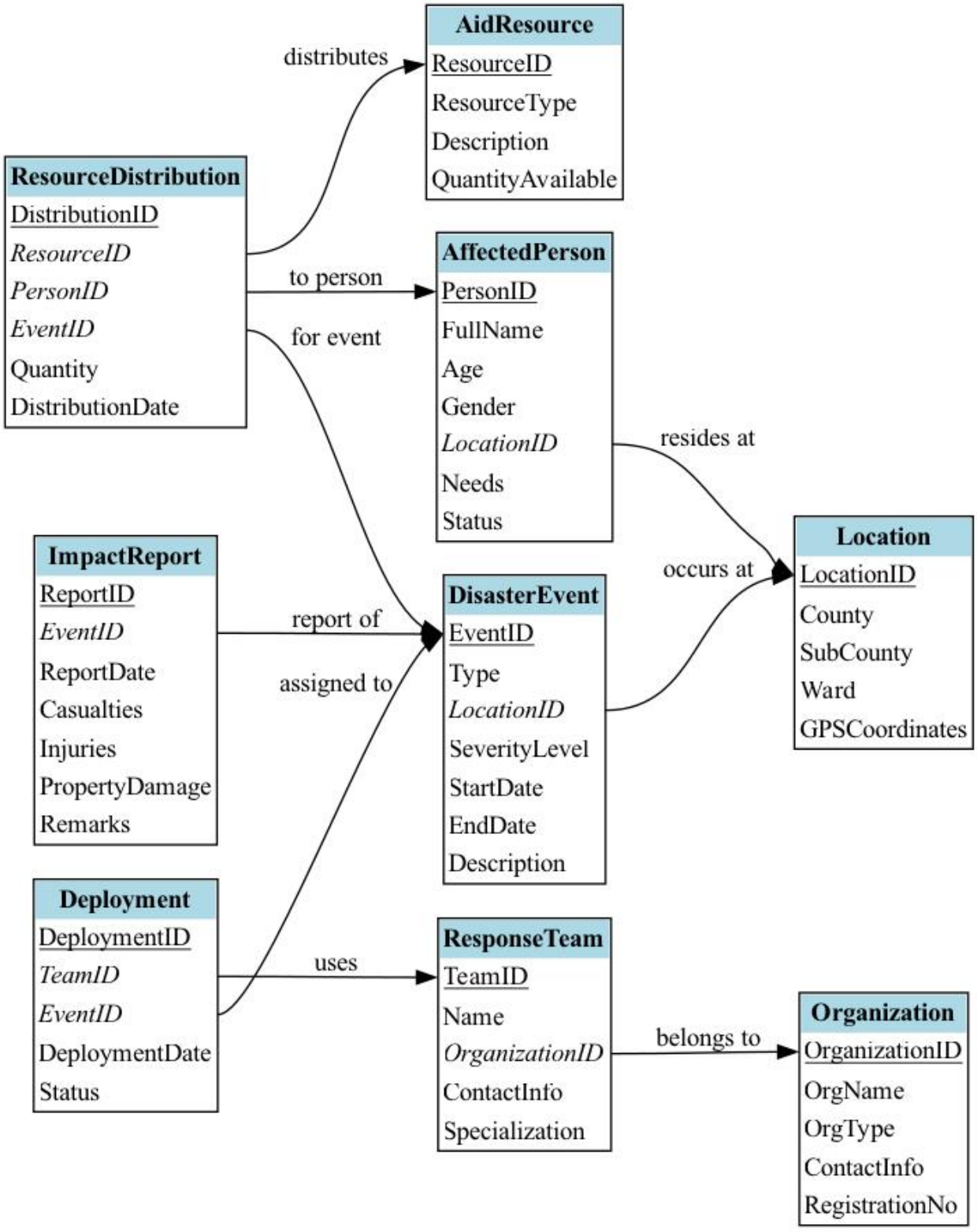
- Maintain records of disaster events (type, date, location, severity) and real-time impacts (e.g., displaced persons)
- Track available emergency response teams and their deployment
- Manage aid resources and their distribution to affected individuals or regions
- Record information about affected individuals or households
- Generate reports and queries to support decision-making
- Monitor NGO/agency interventions to avoid duplication

1.3. Stakeholders

- Government agencies (e.g., NDOC – National Disaster Operations Center) – need structured data for coordination
- NGOs & relief organizations (e.g., Kenya Red Cross, UN) – need access to affected population and aid tracking
- Local affected communities – benefit from faster and more efficient responses via mobile alerts
- Emergency responders – need to track missions and resource usage
- Researchers & policymakers – need historical data for planning and analysis

2. Entity-Relationship Diagram (ERD)

2.1. Design



2.2. Key Entities & Attributes

Entity	Attributes
DisasterEvent	EventID (PK), Type[flood/drought/fire], LocationID (FK), SeverityLevel, StartDate, EndDate, Description
Location	LocationID (PK), County, SubCounty, Ward, GPSCoordinates, risk_level [high/medium/low])
ResponseTeam	TeamID (PK), Name, OrganizationID (FK), ContactInfo, Specialization
Organization	OrganizationID (PK), OrgName, OrgType, ContactInfo, RegistrationNo,Specialization
Deployment	DeploymentID (PK), TeamID (FK), EventID (FK), DeploymentDate, Status
AffectedPerson	PersonID (PK), FullName, Age, Gender, LocationID (FK), Needs, Status
AidResource	ResourceID (PK), ResourceType[food/medicine/tents], Description, QuantityAvailable,Supplier
ResourceDistribution	DistributionID (PK), ResourceID (FK), PersonID (FK), EventID (FK), Quantity, DistributionDate
ImpactReport	ReportID (PK), EventID (FK), ReportDate, Casualties, Injuries, PropertyDamage, Remarks

2.3. Entity Relationships

- One DisasterEvent occurs in one Location (Many-to-One)
  - Explanation: Each disaster event takes place in a single location, but a location can have many disaster events.
- One DisasterEvent can have many Deployment (One-to-Many)
  - Explanation: Each disaster event can have multiple deployments by response teams, but a deployment is always associated with a single disaster event.
- One DisasterEvent can affect many AffectedPersons (via ResourceDistribution or ImpactReports) (One-to-Many)
  - Explanation: Each disaster event can affect many individuals, with affected persons either receiving aid resources or being included in impact reports.
- One DisasterEvent has one or many ImpactReports (One-to-Many)
  - Explanation: A disaster event can have multiple impact reports, but each report pertains to a single disaster event.
- One DisasterEvent can trigger many ResourceDistributions (One-to-Many)
  - Explanation: A disaster event can be the cause of many resource distributions, but each distribution relates to a specific disaster event.
- One Location can be linked to many DisasterEvents (One-to-Many)
  - Explanation: A location can experience multiple disaster events, but each event happens in one location.

- One Location can be linked to many AffectedPersons (One-to-Many)
    - Explanation: A location can have many affected people, but each person resides in a single location.
- One Organization can manage many ResponseTeams (One-to-Many)
    - Explanation: Each organization can manage multiple response teams, but each team belongs to one organization.
- One ResponseTeam belongs to one Organization (Many-to-One)
    - Explanation: Each response team is associated with one organization, but one organization can manage many response teams.
- One ResponseTeam can be deployed to many DisasterEvents (via Deployments) (Many-to-Many through Deployment)
    - Explanation: A response team can be deployed to multiple disaster events, and a disaster event can have many teams deployed to it. This relationship is managed through the deployment table.
- One Deployment links one ResponseTeam to one DisasterEvent (Many-to-One for each FK)
    - Explanation: A deployment is a link between one response team and one disaster event. Multiple deployments can happen between the same teams and events.
- One DisasterEvent can have many Deployments (One-to-Many)
    - Explanation: A disaster event can have many teams deployed to it, with each deployment representing a single team's involvement.
- One ResponseTeam can have many Deployments (One-to-Many)
    - Explanation: A response team can be deployed to many disaster events, with each deployment referring to a specific event.
- One AffectedPerson lives in one Location (Many-to-One)
    - Explanation: Each affected person resides in one location, but each location can have many affected persons.
- One AffectedPerson can receive many AidResources (via ResourceDistribution) (Many-to-Many through ResourceDistribution)
    - Explanation: An affected person can receive many different aid resources, and each resource can be distributed to many people. This is a many-to-many relationship through the ResourceDistribution table.
- One AidResource can be distributed to many AffectedPersons (Many-to-Many through ResourceDistribution)
    - Explanation: A single aid resource (e.g., food, medicine) can be distributed to many affected persons, and each affected person can receive various aid resources.
- One AidResource can be distributed for many DisasterEvents (Many-to-Many through ResourceDistribution)

    - Explanation: Each aid resource can be distributed to multiple disaster events, as resources may be sent to different events over time.
- One AffectedPerson can receive AidResources related to multiple DisasterEvents (Many-to-Many through ResourceDistribution)
    - Explanation: An affected person may receive aid for multiple disaster events, and each event can involve resources being distributed to many affected people.
- One ImpactReport is associated with one DisasterEvent (Many-to-One)
    - Explanation: Each impact report pertains to a specific disaster event, but each event can have multiple impact reports.
- One DisasterEvent can have multiple ImpactReports (One-to-Many)
    - Explanation: A disaster event can have many impact reports, each providing detailed information about the event's aftermath.
- One ResourceDistribution links one AidResource, one AffectedPerson, and one DisasterEvent (Many-to-One for each FK)
    - Explanation: Each resource distribution entry links an aid resource to an affected person for a specific disaster event. This is managed through the ResourceDistribution table.
- One DisasterEvent can have many ResourceDistributions (One-to-Many)
    - Explanation: A disaster event can involve many resource distributions, each linked to different affected persons and resources.
- One AffectedPerson can be involved in many ResourceDistributions (One-to-Many)
    - Explanation: An affected person can receive resources from multiple distributions across different disaster events.
- One AidResource can be part of many ResourceDistributions (One-to-Many)
    - Explanation: A single aid resource (e.g., food or medical supplies) can be distributed to many affected persons across various disaster events.

3. Database Schema

3.1. Normalization

The schema is normalized to Third Normal Form (3NF):

- 1NF: No repeating groups; atomic values
- 2NF: No partial dependencies
- 3NF: No transitive dependencies

3.2. Tables and Fields

Table	Field	Data Type	Constraints

Location	LocationID	INT	PRIMARY KEY
	County	VARCHAR(100)	NOT NULL
	SubCounty	VARCHAR(100)	NOT NULL
	Ward	VARCHAR(100)	
	GPSCoordinates	VARCHAR(255)	
Organization	OrganizationID	INT	PRIMARY KEY
	OrgName	VARCHAR(100)	NOT NULL
	OrgType	VARCHAR(50)	
	ContactInfo	VARCHAR(255)	
	RegistrationNo	VARCHAR(50)	UNIQUE
ResponseTeam	TeamID	INT	PRIMARY KEY
	Name	VARCHAR(100)	NOT NULL
	OrganizationID	INT	FOREIGN KEY → Organization
	ContactInfo	VARCHAR(255)	
	Specialization	VARCHAR(100)	
DisasterEvent	EventID	INT	PRIMARY KEY
	Type	VARCHAR(100)	NOT NULL
	LocationID	INT	FOREIGN KEY → Location
	SeverityLevel	VARCHAR(50)	
	StartDate	DATE	
	EndDate	DATE	
	Description	TEXT	
ImpactReport	ReportID	INT	PRIMARY KEY
	EventID	INT	FOREIGN KEY → DisasterEvent
	ReportDate	DATE	
	Casualties	INT	
	Injuries	INT	
	PropertyDamage	TEXT	
	Remarks	TEXT	
Deployment	DeploymentID	INT	PRIMARY KEY
	TeamID	INT	FOREIGN KEY → ResponseTeam
	EventID	INT	FOREIGN KEY → DisasterEvent
	DeploymentDate	DATE	
	Status	VARCHAR(50)	
AffectedPerson	PersonID	INT	PRIMARY KEY
	FullName	VARCHAR(100)	NOT NULL
	Age	INT	

	Gender	VARCHAR(10)	
	LocationID	INT	FOREIGN KEY → Location
	Needs	TEXT	
	Status	VARCHAR(50)	
AidResource	ResourceID	INT	PRIMARY KEY
	ResourceType	VARCHAR(100)	NOT NULL
	Description	TEXT	
	QuantityAvailable	INT	DEFAULT 0
ResourceDistribution	DistributionID	INT	PRIMARY KEY
	ResourceID	INT	FOREIGN KEY → AidResource
	PersonID	INT	FOREIGN KEY → AffectedPerson
	EventID	INT	FOREIGN KEY → DisasterEvent
	Quantity	INT	NOT NULL
	DistributionDate	DATE	

3.3. Indexes (Performance Optimization)

Table	Field(s) to Index	Purpose
DisasterEvent	LocationID, SeverityLevel	Faster filtering by location or severity
ResponseTeam	OrganizationID	Quick lookup of teams under an organization
AffectedPerson	LocationID, Status	For filtering affected people by location/status
ResourceDistribution	EventID, PersonID, ResourceID	Faster joins and reports
Deployment	TeamID, EventID	Efficient team-event deployment tracking

4. SQL Code Implementation

4.1. Database & Table Creation

```
-- Create the database
CREATE DATABASE DisasterResponseDB;
USE DisasterResponseDB;


-- Location table
CREATE TABLE Location (
```

```
    LocationID INT PRIMARY KEY,
    County VARCHAR(100),
    SubCounty VARCHAR(100),
    Ward VARCHAR(100),
    GPSCoordinates VARCHAR(255)
);

-- Organization table
CREATE TABLE Organization (
    OrganizationID INT PRIMARY KEY,
    OrgName VARCHAR(100) NOT NULL,
    OrgType VARCHAR(50),
    ContactInfo VARCHAR(255),
    RegistrationNo VARCHAR(50) UNIQUE
);

-- ResponseTeam table
CREATE TABLE ResponseTeam (
    TeamID INT PRIMARY KEY,
    Name VARCHAR(100),
    OrganizationID INT,
    ContactInfo VARCHAR(255),
    Specialization VARCHAR(100),
    FOREIGN KEY (OrganizationID) REFERENCES Organization(OrganizationID)
);

-- DisasterEvent table
CREATE TABLE DisasterEvent (
    EventID INT PRIMARY KEY,
    Type VARCHAR(100),
    LocationID INT,
    SeverityLevel VARCHAR(50),
    StartDate DATE,
    EndDate DATE,
    Description TEXT,
    FOREIGN KEY (LocationID) REFERENCES Location(LocationID)
);

-- ImpactReport table
CREATE TABLE ImpactReport (
```

```
    ReportID INT PRIMARY KEY,
    EventID INT,
    ReportDate DATE,
    Casualties INT,
    Injuries INT,
    PropertyDamage TEXT,
    Remarks TEXT,
    FOREIGN KEY (EventID) REFERENCES DisasterEvent(EventID)
);

-- Deployment table
CREATE TABLE Deployment (
    DeploymentID INT PRIMARY KEY,
    TeamID INT,
    EventID INT,
    DeploymentDate DATE,
    Status VARCHAR(50),
    FOREIGN KEY (TeamID) REFERENCES ResponseTeam(TeamID),
    FOREIGN KEY (EventID) REFERENCES DisasterEvent(EventID)
);

-- AffectedPerson table
CREATE TABLE AffectedPerson (
    PersonID INT PRIMARY KEY,
    FullName VARCHAR(100),
    Age INT,
    Gender VARCHAR(10),
    LocationID INT,
    Needs TEXT,
    Status VARCHAR(50),
    FOREIGN KEY (LocationID) REFERENCES Location(LocationID)
);

-- AidResource table
CREATE TABLE AidResource (
    ResourceID INT PRIMARY KEY,
    ResourceType VARCHAR(100),
    Description TEXT,
    QuantityAvailable INT
);
```

```
-- ResourceDistribution table
CREATE TABLE ResourceDistribution (
   DistributionID INT PRIMARY KEY,
   ResourceID INT,
   PersonID INT,
   EventID INT,
   Quantity INT,
   DistributionDate DATE,
   FOREIGN KEY (ResourceID) REFERENCES AidResource(ResourceID),
   FOREIGN KEY (PersonID) REFERENCES AffectedPerson(PersonID),
   FOREIGN KEY (EventID) REFERENCES DisasterEvent(EventID)
);
```

4.2. Data Manupulation (CRUD Operations)

4.2.1. Insert Example (Create)

```
INSERT INTO Location VALUES (1, 'Nairobi', 'Westlands', 'Kangemi', '1.2921°S, 36.8219°E');
INSERT INTO Organization VALUES (1, 'Red Cross Kenya', 'NGO', 'contact@redcross.or.ke',
'REG12345');
INSERT INTO ResponseTeam VALUES (1, 'Alpha Team', 1, '0722123456', 'Medical Aid');
```

4.2.2. Read Example (Select)

```
SELECT * FROM ResponseTeam WHERE Specialization = 'Medical Aid';
```

4.2.3. Update Example

```
UPDATE AffectedPerson
SET Status = 'Relocated'
WHERE PersonID = 1;
```

4.2.4. Delete Example

```
DELETE FROM Deployment WHERE DeploymentID = 3;
```

4.3. Advanced Queries

4.3.1. Get all teams deployed to a specific disaster

```
-- Join: List all deployments with team and event info
SELECT RT.Name, D.DeploymentDate
```

```
FROM Deployment D
JOIN ResponseTeam RT ON D.TeamID = RT.TeamID
WHERE D.EventID = 1;
```

4.3.2. Total aid distributed per disaster

```
-- Aggregate: Total aid distributed per disaster event
SELECT E.EventID, E.Type, SUM(RD.Quantity) AS TotalAid
FROM ResourceDistribution RD
JOIN DisasterEvent E ON RD.EventID = E.EventID
GROUP BY E.EventID, E.Type;
```

4.3.3. List affected people and aid received

```
SELECT AP.FullName, AR.ResourceType, RD.Quantity
FROM ResourceDistribution RD
JOIN AffectedPerson AP ON RD.PersonID = AP.PersonID
JOIN AidResource AR ON RD.ResourceID = AR.ResourceID;
```

4.3.4. Subquery: Events with more than 100 casualties

```
SELECT * FROM DisasterEvent
WHERE EventID IN
( SELECT EventID FROM ImpactReport WHERE Casualties > 100 );
```