INTERNATIONAL ISLAMIC UNIVERSITY CHITTAGONG

PROJECT REPORT

Project Title : Donation Management System

Course Title : Database Management System Lab

Course Code : CSE 2424

Semester : 4BF(Spring-2025)

Submitted By

- Name: Shakila Jahan Saima
- Student ID: C233442

Instructor

Ms. Mysha Sarin Kabisha
Assistant Lecturer
Department of Computer Science and Engineering
International Islamic University Chittagong

Submission Date: 13.07.2025

Abstract

The Donation Management System is designed to streamline the process of collecting, managing, and distributing charitable donations. The main objective of the system is to ensure transparency and efficiency in donor contributions, campaign management, and beneficiary support. The problem domain focuses on the challenges faced by charitable organizations in tracking donations, ensuring proper allocation to beneficiaries, and maintaining accurate donor records.

The database is structured using a relational model, capturing entities such as Donors, Donations, Beneficiaries, Distributions, Campaigns, and Admins. Relationships are established to represent real-world interactions, including how donations are made by donors, how funds are distributed to beneficiaries through specific campaigns, and how the entire process is administered. Primary keys (PK) and foreign keys (FK) are clearly defined to maintain referential integrity across the system.

The project was developed using Oracle SQL for database design, with Entity-Relationship Diagrams (ERDs) used for visual schema representation. Tools such as SQL Developer or similar database platforms may be employed for implementation and testing.

Key outcomes of this system include improved data management for donation tracking, better visibility into donor and beneficiary interactions, and a scalable framework that can support future enhancements such as automated reporting and audit tracking.

Table of Contents :

1.Abstract—-- 2

2.Table of Contents—--- 3

3.Introduction—-- 4

4.Requirement Analysis—-- 5

5.Conceptual Model (ERD)-- 7

6.Normalization Process—--- 9

7. Final Relational Schema—--- 12

8.Table Creation and Sample Data—--- 14

9.User Interface Design—-- 22

10.DML Queries—-- 25

11.Challenges Faced—--- 32

12.Conclusion—--- 33

13.References—--- 34

3.Introduction

3.1 Background

The field of donation and charity management plays a crucial role in supporting underprivileged communities, disaster relief, and various social causes. However, many organizations still rely on manual methods or scattered systems to manage donations, donors, beneficiaries, and campaigns, leading to inefficiencies, data loss, and a lack of transparency. A well-designed database solution can address these challenges by offering a centralized and organized way to track contributions, manage donor and beneficiary information, and monitor the distribution of resources. Database-driven systems provide consistency, integrity, and scalability, enabling non-profits and charitable organizations to operate more effectively and maintain public trust.

3.2 Objective

The primary objective of this project is to design and implement a relational database system that facilitates the efficient management of donation activities. This includes capturing donor information, recording donation transactions, managing campaign data, tracking beneficiary needs, and documenting distribution records. The system aims to ensure transparency, improve data accuracy, and support administrative decision-making through well-structured data storage and retrieval mechanisms.

3.3 Scope

Inclusions:

- Recording donor profiles and contact information.

- Capturing donation details, including payment methods and purposes.

- Managing campaigns with timelines and fundraising goals.

- Storing beneficiary details and their associated needs.

- Tracking the distribution of donations to beneficiaries.
- Assigning administrative roles for system management.

Exclusions:

- Development of a full-fledged user interface or web portal.
- Integration with external payment gateways.
- Real-time notification or reporting features beyond database-level functionalities.
- Advanced analytics or AI-based decision support systems.

4. Requirement Analysis

4.1 Functional Requirements

The system must support the following core functionalities:

- **Donor Registration**: Add and manage donor information (name, contact, address).
- **Record Donations**: Log donation entries with amount, date, payment method, and purpose.
- **Beneficiary Management**: Register beneficiaries, including their contact details and need descriptions.
- **Campaign Management**: Create and manage campaigns with start/end dates and target amounts.

- **Distribution Tracking**: Record the distribution of funds or resources from donations to beneficiaries.
- **Admin Operations**: Allow authorized admins to manage all system data securely.
- **View Reports**: Generate reports based on donations, campaigns, and distributions.

4.2 Non-Functional Requirements

- **Performance:** The system should handle large volumes of data without performance degradation.
- **Reliability**: Data must be stored and retrieved reliably with transactional integrity.
- **Usability**: The system should have a user-friendly interface for admins and support staff.
- **Security**: Admin-level operations should be protected with authentication credentials.
- **Scalability**: The system should allow future expansion, such as integrating online donations or mobile access.
- **Maintainability**: The database schema should be well-documented and easy to maintain or upgrade.

5. Conceptual Model (ERD)

The conceptual model of the Donation Management System is represented using an **Entity-Relationship Diagram (ERD)**. It includes the following key entities and their relationships:

- **Donor**: Holds donor details.
- **Donation**: Linked to Donor; includes amount, date, payment method, and purpose.
- **Beneficiary**: Contains data on individuals or groups receiving aid.
- **Distribution**: Links donations to beneficiaries; tracks how and when aid is given.
- **Campaign**: Represents fundraising efforts; linked to donations.
- **Admin**: Users who manage the system's data and operations.

Each entity includes primary keys (PK) and relevant attributes, while relationships use foreign keys (FK) to ensure data integrity. The ERD diagram (attached as a PDF) visually illustrates the structure and cardinality between entities, such as one-to-many and many-to-many relationships.

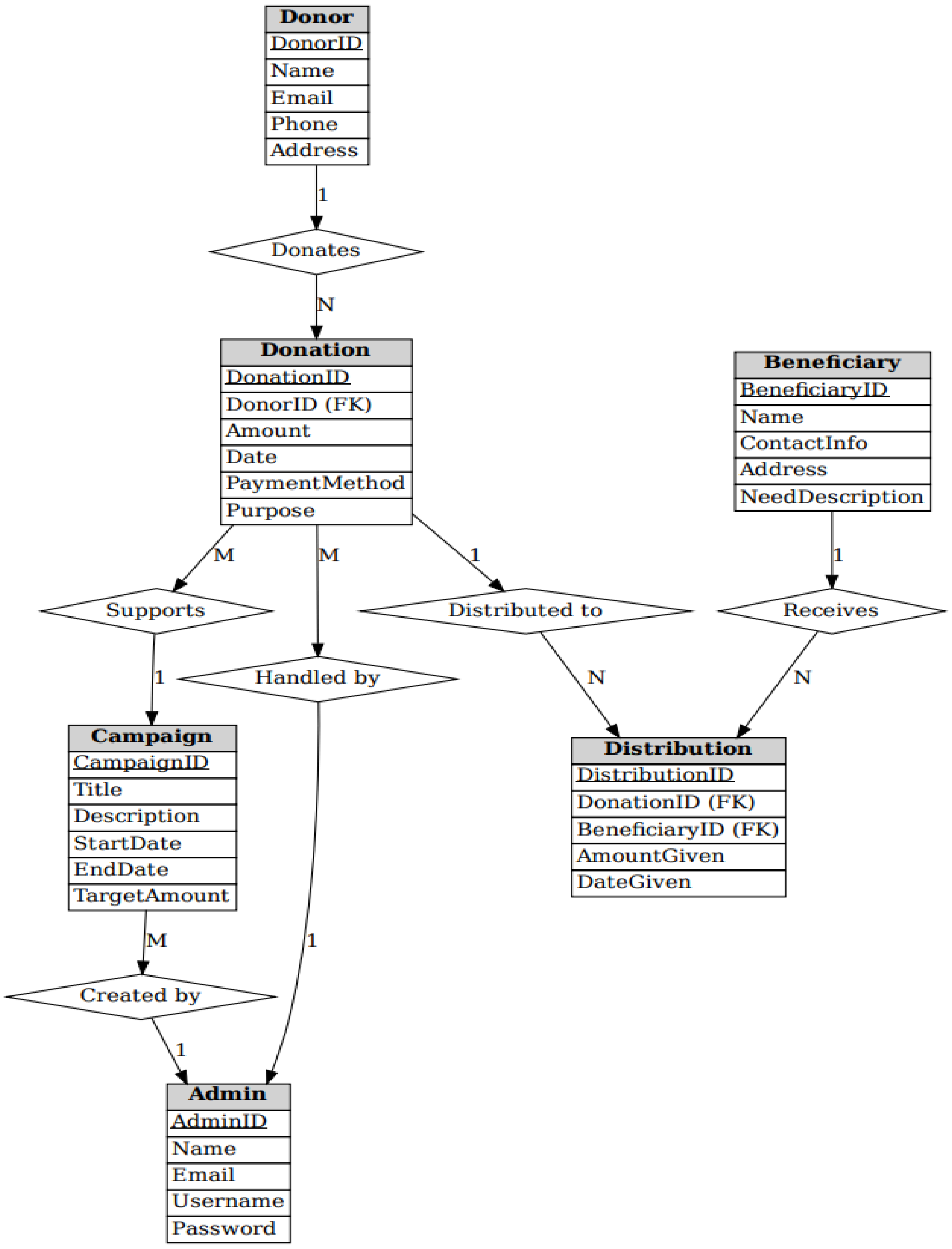


Figure 1: Entity-Relationship Diagram of the Donation Management System

6. Normalization Process

Normalization is the process of organizing data in a database to minimize redundancy and improve data integrity. It involves decomposing larger, unstructured tables into smaller, well-defined tables using normal forms. For the Donation Management System, the data undergoes the following normalization stages

Step1 :

Unnormalized Form (UNF)

In the unnormalized form, all information is stored in a single large record. This record might include details about the donor, multiple donations made by the donor, the beneficiaries who received the donations, campaign details, and the admin who handled the distribution.
For example, a single row might contain:

- Donor ID and Name
- Donation ID, Amount, Payment Method, Purpose
- Beneficiary Name, Need Description
- Campaign Title and Dates
- Admin Name and Email

This structure has multiple problems:

- Repeating groups (e.g., multiple donations or beneficiaries in one record)
- Redundant data (same donor or admin repeated across records)
- Inconsistency and difficulty in updates

Step 2:

First Normal Form (1NF)

To bring the data into First Normal Form, we eliminate repeating groups and ensure all values are atomic (indivisible).
Actions taken:

- Separate each multi-valued or repeating group into its own row.
- Identify unique entities such as Donor, Donation, Beneficiary, Campaign, and Admin.
- Assign a unique identifier (primary key) to each entity.

After 1NF, each table stores a single type of entity, but dependencies between attributes might still exist.

Step 3:

Second Normal Form (2NF)

In this step, we remove **partial dependencies**—which means we ensure that every non-key attribute is fully dependent on the whole primary key.
Actions taken:

- If a table has a composite primary key, and a non-key attribute depends on only part of it, we move that attribute to a separate table.
- For instance, in a Donation table linked to a Donor, attributes like Donor Name and Email are moved to a separate Donor table since they depend only on DonorID.
- Similarly, Campaign information is separated into its own table, since campaign details are not functionally dependent on donation or beneficiary IDs.
  This step ensures that every table now has attributes fully dependent on their complete primary key.

Step 4:

Third Normal Form (3NF)

In the final step, we remove **transitive dependencies**—which occur when non-key attributes depend on other non-key attributes rather than directly on the primary key.
Actions taken:

- Move Admin information (such as name, email, and login details) into a separate Admin table instead of storing them in Campaign or Distribution tables.

- Ensure that all non-key attributes in every table depend **only** on the primary key of that table and not on other non-key attributes.

Now, the structure is clean, fully normalized, and ensures maximum data consistency and flexibility.

Final Schema in 3NF

1.Donor(DonorID, Name, Email, Phone, Address)

2.Donation(DonationID, DonorID [FK], Amount, Date, PaymentMethod, Purpose)

3.Beneficiary(BeneficiaryID, Name, ContactInfo, Address, NeedDescription)

4.Campaign(CampaignID, Title, Description, StartDate, EndDate, TargetAmount)

5.Admin(AdminID, Name, Email, Username, Password)

6.Distribution(DistributionID, DonationID [FK], BeneficiaryID [FK], AmountGiven, DateGiven, CampaignID [FK], AdminID [FK])

7. Final Relational Schema

1. Donor

Donor(DonorID, Name, Email, Phone, Address)

- **PK**: DonorID
- **Constraints**:
  - Email is unique
  - Phone is unique

2. Donation

Donation(DonationID, DonorID, Amount, Date, PaymentMethod, Purpose)

- **PK**: DonationID
- **FK**: DonorID → Donor(DonorID)
- **Constraints**:
  - PaymentMethod must be one of (e.g., 'Cash', 'Card', 'Mobile')
  - Amount > 0

3. Beneficiary

Beneficiary(BeneficiaryID, Name, ContactInfo, Address, NeedDescription)

- **PK**: BeneficiaryID
- **Constraints**:
  - ContactInfo is unique

4. Campaign

Campaign(CampaignID, Title, Description, StartDate, EndDate, TargetAmount)

- **PK**: CampaignID
- **Constraints**:
  - TargetAmount ≥ 0
  - StartDate < EndDate
  - Title is unique

5. Admin

Admin(AdminID, Name, Email, Username, Password)

- **PK**: AdminID
- **Constraints**:
  - Username is unique
  - Email is unique
  - Password must meet security requirements

6. Distribution

Distribution(DistributionID, DonationID, BeneficiaryID, AmountGiven, DateGiven, CampaignID, AdminID)

- **PK**: DistributionID
- **FKs**:
  - DonationID → Donation(DonationID)
  - BeneficiaryID → Beneficiary(BeneficiaryID)

  - CampaignID → Campaign(CampaignID)
  - AdminID → Admin(AdminID)
- **Constraints**:
  - AmountGiven ≤ Donation.Amount
  - DateGiven ≥ Donation.Date

8. Table Creation and Sample Data

8.1 Table Structures (DDL) :

Donor Table

```
CREATE TABLE Donor (
   DonorID NUMBER PRIMARY KEY,
   Name VARCHAR2(100),
   Email VARCHAR2(100),
   Phone VARCHAR2(20),
   Address VARCHAR2(255)
);
```

Donation Table

```
CREATE TABLE Donation (
   DonationID NUMBER PRIMARY KEY,
   DonorID NUMBER,
   Amount NUMBER(12, 2),
   DonationDate DATE,
   PaymentMethod VARCHAR2(50),
   Purpose VARCHAR2(4000),
   FOREIGN KEY (DonorID) REFERENCES Donor(DonorID)
```

Campaign Table

```
CREATE TABLE Campaign (
   CampaignID NUMBER PRIMARY KEY,
   Title VARCHAR2(100),
   Description  VARCHAR2(100),
   StartDate DATE,
   EndDate DATE,
   TargetAmount NUMBER(12, 2)
);
```

Admin Table

```
CREATE TABLE Admin (
   AdminID NUMBER PRIMARY KEY,
   Name VARCHAR2(100),
   Email VARCHAR2(100),
   Username VARCHAR2(50),
   Password VARCHAR2(100)
);
```

Beneficiary Table

```
CREATE TABLE Beneficiary (
   BeneficiaryID NUMBER PRIMARY KEY,
   Name VARCHAR2(100),
   ContactInfo VARCHAR2(100),
   Address VARCHAR2(255),
   NeedDescription  VARCHAR2(100),
);
```

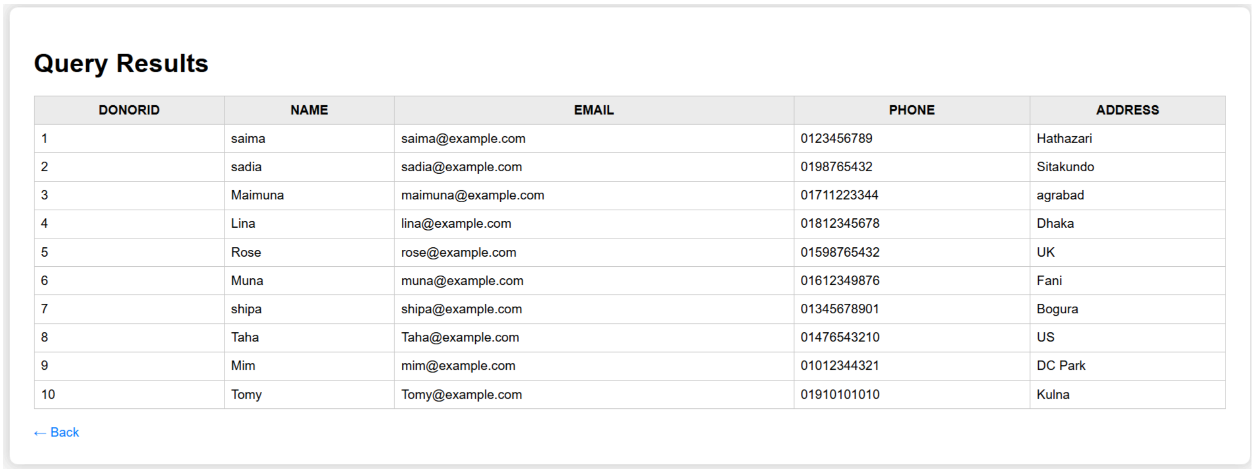
Distribution

```
CREATE TABLE Distribution (
   DistributionID NUMBER PRIMARY KEY,
   DonationID NUMBER,
   BeneficiaryID NUMBER,
    AmountGiven NUMBER(12, 2),
    DateGiven DATE,
    FOREIGN KEY (DonationID) REFERENCES Donation(DonationID),
    FOREIGN KEY (BeneficiaryID) REFERENCES Beneficiary(BeneficiaryID)
);
```

8.2 Sample Data (DML) :

Donor Table

```
INSERT ALL
    INTO Donor VALUES (1, 'saima', 'saima@example.com', '0123456789',
'Hathazari')
    INTO Donor VALUES (2, 'sadia', 'sadia@example.com', '0198765432',
'Sitakundo')
    INTO Donor VALUES (3, 'Maimuna', 'maimuna@example.com',
'01711223344', 'agrabad')
    INTO Donor VALUES (4, 'Lina', 'lina@example.com', '01812345678', 'Dhaka')
    INTO Donor VALUES (5, 'Rose', 'rose@example.com', '01598765432', 'UK')
    INTO Donor VALUES (6, 'Muna', 'muna@example.com', '01612349876', 'Fani')
    INTO Donor VALUES (7, 'shipa', 'shipa@example.com', '01345678901',
'Bogura')
    INTO Donor VALUES (8, 'Taha', 'Taha@example.com', '01476543210', 'US')
    INTO Donor VALUES (9, 'Mim', 'mim@example.com', '01012344321', 'DC
Park')
    INTO Donor VALUES (10, 'Tomy', 'Tomy@example.com', '01910101010',
'Kulna')
SELECT * FROM dual;
```



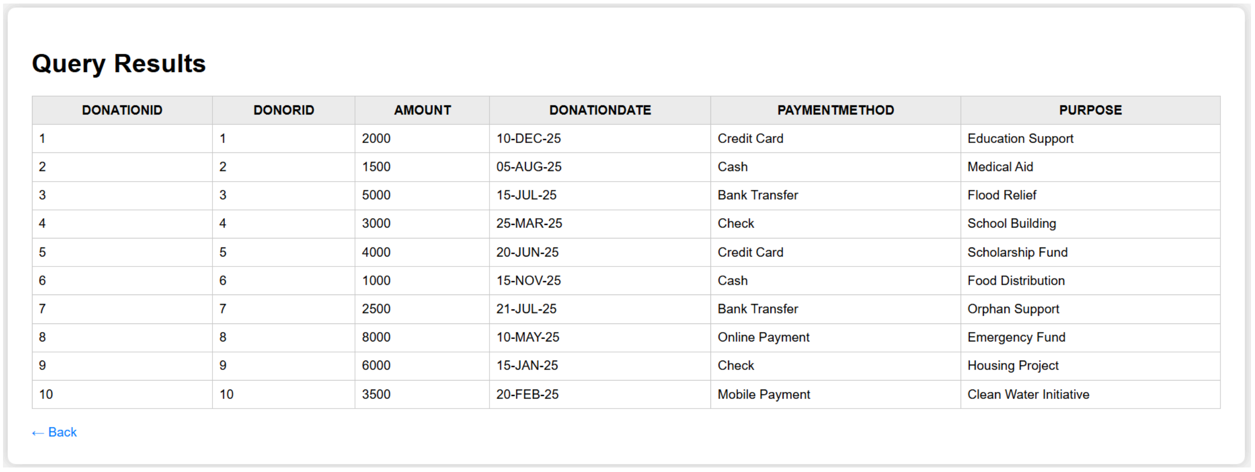
Query Results

DONORID	NAME	EMAIL	PHONE	ADDRESS
1	saima	saima@example.com	0123456789	Hathazari
2	sadia	sadia@example.com	0198765432	Sitakundo
3	Maimuna	maimuna@example.com	01711223344	agrabad
4	Lina	lina@example.com	01812345678	Dhaka
5	Rose	rose@example.com	01598765432	UK
6	Muna	muna@example.com	01612349876	Fani
7	shipa	shipa@example.com	01345678901	Bogura
8	Taha	Taha@example.com	01476543210	US
9	Mim	mim@example.com	01012344321	DC Park
10	Tomy	Tomy@example.com	01910101010	Kulna

← Back

Donation Table

```
INSERT ALL
   INTO Donation
      VALUES (1, 1, 2000.00, TO_DATE('2025-12-10', 'YYYY-MM-DD'), 'Credit
Card', 'Education Support')
   INTO Donation
      VALUES (2, 2, 1500.00, TO_DATE('2025-08-05', 'YYYY-MM-DD'), 'Cash',
'Medical Aid')
   INTO Donation
      VALUES (3, 3, 5000.00, TO_DATE('2025-07-15', 'YYYY-MM-DD'), 'Bank
Transfer', 'Flood Relief')
   INTO Donation
      VALUES (4, 4, 3000.00, TO_DATE('2025-03-25', 'YYYY-MM-DD'), 'Check',
'School Building')
   INTO Donation
      VALUES (5, 5, 4000.00, TO_DATE('2025-06-20', 'YYYY-MM-DD'), 'Credit
Card', 'Scholarship Fund')
   INTO Donation
      VALUES (6, 6, 1000.00, TO_DATE('2025-11-15', 'YYYY-MM-DD'), 'Cash',
'Food Distribution')
   INTO Donation
      VALUES (7, 7, 2500.00, TO_DATE('2025-07-21', 'YYYY-MM-DD'), 'Bank
Transfer', 'Orphan Support')
   INTO Donation
      VALUES (8, 8, 8000.00, TO_DATE('2025-05-10', 'YYYY-MM-DD'), 'Online
Payment', 'Emergency Fund')
   INTO Donation
      VALUES (9, 9, 6000.00, TO_DATE('2025-01-15', 'YYYY-MM-DD'), 'Check',
'Housing Project')
   INTO Donation
      VALUES (10, 10, 3500.00, TO_DATE('2025-02-20', 'YYYY-MM-DD'), 'Mobile
Payment', 'Clean Water Initiative')
SELECT * FROM dual;
```

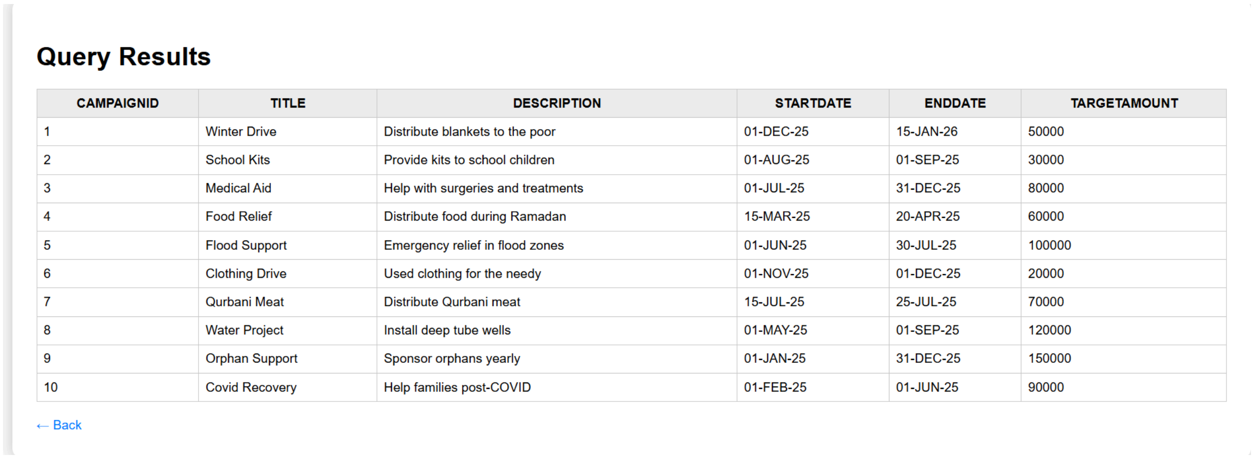

Query Results

DONATIONID	DONORID	AMOUNT	DONATIONDATE	PAYMENTMETHOD	PURPOSE
1	1	2000	10-DEC-25	Credit Card	Education Support
2	2	1500	05-AUG-25	Cash	Medical Aid
3	3	5000	15-JUL-25	Bank Transfer	Flood Relief
4	4	3000	25-MAR-25	Check	School Building
5	5	4000	20-JUN-25	Credit Card	Scholarship Fund
6	6	1000	15-NOV-25	Cash	Food Distribution
7	7	2500	21-JUL-25	Bank Transfer	Orphan Support
8	8	8000	10-MAY-25	Online Payment	Emergency Fund
9	9	6000	15-JAN-25	Check	Housing Project
10	10	3500	20-FEB-25	Mobile Payment	Clean Water Initiative

Campaign Table

```
INSERT ALL
    INTO Campaign
    VALUES (1, 'Winter Drive', 'Distribute blankets to the poor',
TO_DATE('2025-12-01', 'YYYY-MM-DD'), TO_DATE('2026-01-15',
'YYYY-MM-DD'), 50000)
    INTO Campaign
    VALUES (2, 'School Kits', 'Provide kits to school children',
TO_DATE('2025-08-01', 'YYYY-MM-DD'), TO_DATE('2025-09-01',
'YYYY-MM-DD'), 30000)
    INTO Campaign
    VALUES (3, 'Medical Aid', 'Help with surgeries and treatments',
TO_DATE('2025-07-01', 'YYYY-MM-DD'), TO_DATE('2025-12-31',
'YYYY-MM-DD'), 80000)
    INTO Campaign
    VALUES (4, 'Food Relief', 'Distribute food during Ramadan',
TO_DATE('2025-03-15', 'YYYY-MM-DD'), TO_DATE('2025-04-20',
'YYYY-MM-DD'), 60000)
    INTO Campaign
    VALUES (5, 'Flood Support', 'Emergency relief in flood zones',
TO_DATE('2025-06-01', 'YYYY-MM-DD'), TO_DATE('2025-07-30',
'YYYY-MM-DD'), 100000)
    INTO Campaign
    VALUES (6, 'Clothing Drive', 'Used clothing for the needy',
TO_DATE('2025-11-01', 'YYYY-MM-DD'), TO_DATE('2025-12-01',
'YYYY-MM-DD'), 20000)
    INTO Campaign
    VALUES (7, 'Qurbani Meat', 'Distribute Qurbani meat', TO_DATE('2025-07-15',
'YYYY-MM-DD'), TO_DATE('2025-07-25', 'YYYY-MM-DD'), 70000)
```

```
    INTO Campaign
    VALUES (8, 'Water Project', 'Install deep tube wells', TO_DATE('2025-05-01',
'YYYY-MM-DD'), TO_DATE('2025-09-01', 'YYYY-MM-DD'), 120000)
    INTO Campaign
    VALUES (9, 'Orphan Support', 'Sponsor orphans yearly',
TO_DATE('2025-01-01', 'YYYY-MM-DD'), TO_DATE('2025-12-31',
'YYYY-MM-DD'), 150000)
    INTO Campaign
    VALUES (10, 'Covid Recovery', 'Help families post-COVID',
TO_DATE('2025-02-01', 'YYYY-MM-DD'), TO_DATE('2025-06-01',
'YYYY-MM-DD'), 90000)
SELECT * FROM dual;
```



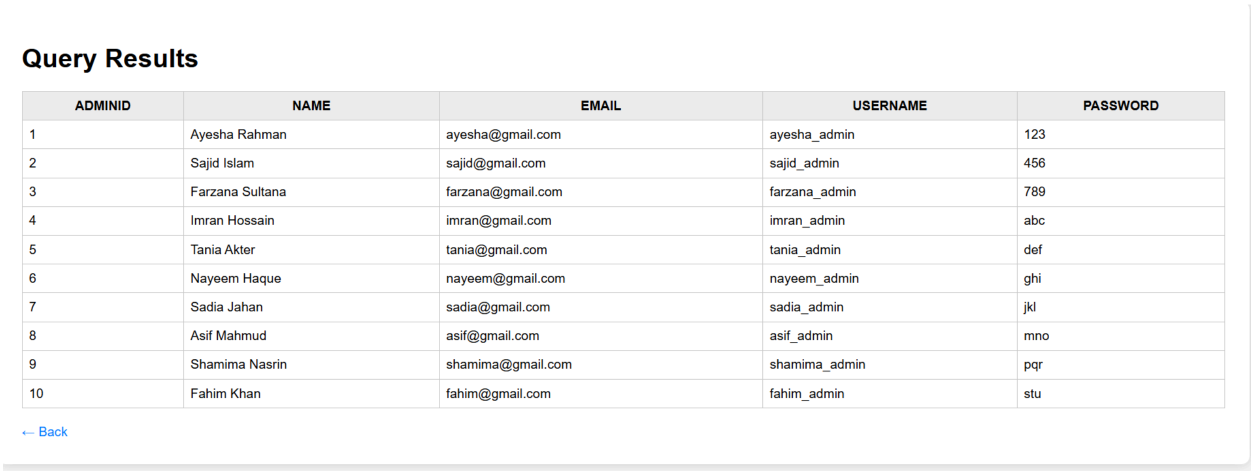
Query Results

CAMPAIGNID	TITLE	DESCRIPTION	STARTDATE	ENDDATE	TARGETAMOUNT
1	Winter Drive	Distribute blankets to the poor	01-DEC-25	15-JAN-26	50000
2	School Kits	Provide kits to school children	01-AUG-25	01-SEP-25	30000
3	Medical Aid	Help with surgeries and treatments	01-JUL-25	31-DEC-25	80000
4	Food Relief	Distribute food during Ramadan	15-MAR-25	20-APR-25	60000
5	Flood Support	Emergency relief in flood zones	01-JUN-25	30-JUL-25	100000
6	Clothing Drive	Used clothing for the needy	01-NOV-25	01-DEC-25	20000
7	Qurbani Meat	Distribute Qurbani meat	15-JUL-25	25-JUL-25	70000
8	Water Project	Install deep tube wells	01-MAY-25	01-SEP-25	120000
9	Orphan Support	Sponsor orphans yearly	01-JAN-25	31-DEC-25	150000
10	Covid Recovery	Help families post-COVID	01-FEB-25	01-JUN-25	90000

Admin Table

```
INSERT ALL
    INTO Admin VALUES (1, 'Ayesha Rahman', 'ayesha@gmail.com',
'ayesha_admin', '123')
    INTO Admin VALUES (2, 'Sajid Islam', 'sajid@gmail.com', 'sajid_admin', '456')
    INTO Admin VALUES (3, 'Farzana Sultana', 'farzana@gmail.com',
'farzana_admin', '789')
    INTO Admin VALUES (4, 'Imran Hossain', 'imran@gmail.com', 'imran_admin',
'abc')
    INTO Admin VALUES (5, 'Tania Akter', 'tania@gmail.com', 'tania_admin', 'def')
    INTO Admin VALUES (6, 'Nayeem Haque', 'nayeem@gmail.com',
'nayeem_admin', 'ghi')
    INTO Admin VALUES (7, 'Sadia Jahan', 'sadia@gmail.com', 'sadia_admin',
'jkl')
    INTO Admin VALUES (8, 'Asif Mahmud', 'asif@gmail.com', 'asif_admin', 'mno')
```

```
   INTO Admin VALUES (9, 'Shamima Nasrin', 'shamima@gmail.com',
'shamima_admin', 'pqr')
   INTO Admin VALUES (10, 'Fahim Khan', 'fahim@gmail.com', 'fahim_admin',
'stu')
SELECT * FROM dual;
```



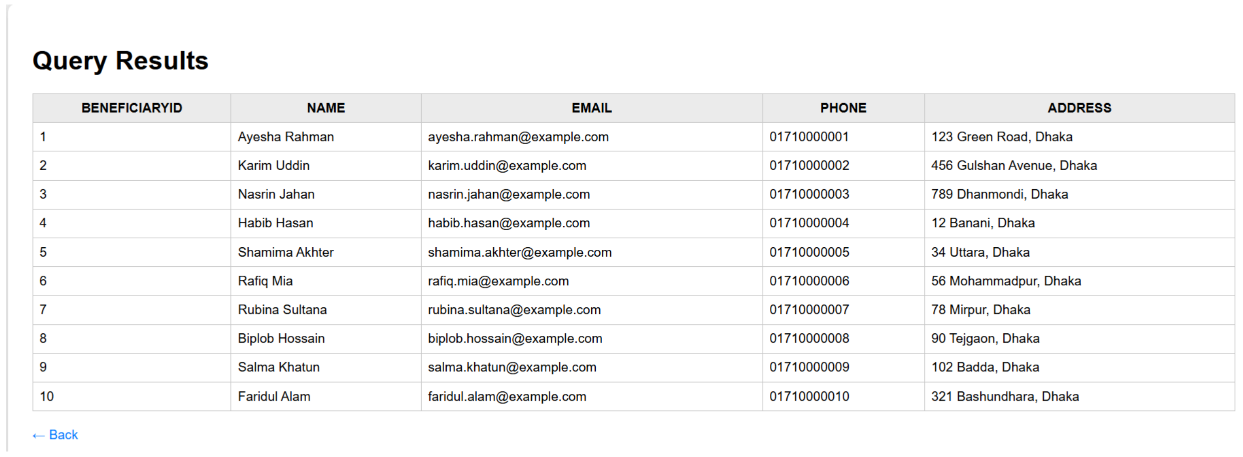
Query Results

ADMINID	NAME	EMAIL	USERNAME	PASSWORD
1	Ayesha Rahman	ayesha@gmail.com	ayesha_admin	123
2	Sajid Islam	sajid@gmail.com	sajid_admin	456
3	Farzana Sultana	farzana@gmail.com	farzana_admin	789
4	Imran Hossain	imran@gmail.com	imran_admin	abc
5	Tania Akter	tania@gmail.com	tania_admin	def
6	Nayeem Haque	nayeem@gmail.com	nayeem_admin	ghi
7	Sadia Jahan	sadia@gmail.com	sadia_admin	jkl
8	Asif Mahmud	asif@gmail.com	asif_admin	mno
9	Shamima Nasrin	shamima@gmail.com	shamima_admin	pqr
10	Fahim Khan	fahim@gmail.com	fahim_admin	stu

← Back

Beneficiary Table

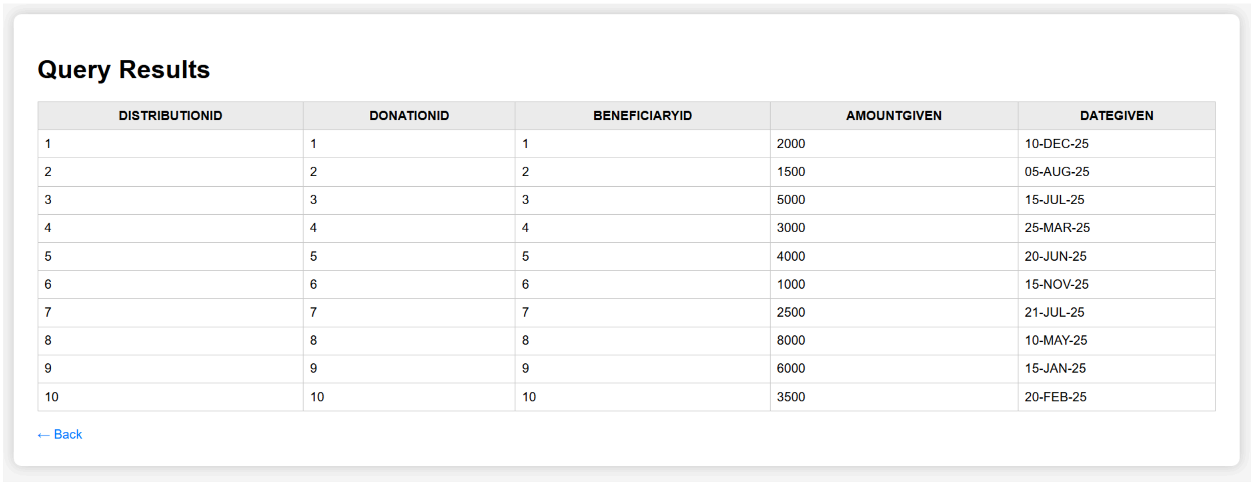
```
INSERT ALL
 INTO Beneficiary  VALUES(1, 'Ayesha Rahman',
'ayesha.rahman@example.com', '01710000001', '123 Green Road, Dhaka'),
 INTO Beneficiary  VALUES(2, 'Karim Uddin', 'karim.uddin@example.com',
'01710000002', '456 Gulshan Avenue, Dhaka'),
 INTO Beneficiary  VALUES(3, 'Nasrin Jahan', 'nasrin.jahan@example.com',
'01710000003', '789 Dhanmondi, Dhaka'),
 INTO Beneficiary  VALUES(4, 'Habib Hasan', 'habib.hasan@example.com',
'01710000004', '12 Banani, Dhaka'),
 INTO Beneficiary  VALUES(5, 'Shamima Akhter',
'shamima.akhter@example.com', '01710000005', '34 Uttara, Dhaka'),
 INTO Beneficiary  VALUES(6, 'Rafiq Mia', 'rafiq.mia@example.com',
'01710000006', '56 Mohammadpur, Dhaka'),
 INTO Beneficiary  VALUES(7, 'Rubina Sultana', 'rubina.sultana@example.com',
'01710000007', '78 Mirpur, Dhaka'),
 INTO Beneficiary  VALUES(8, 'Biplob Hossain', 'biplob.hossain@example.com',
'01710000008', '90 Tejgaon, Dhaka'),
 INTO Beneficiary  VALUES(9, 'Salma Khatun', 'salma.khatun@example.com',
'01710000009', '102 Badda, Dhaka'),
 INTO Beneficiary  VALUES(10, 'Faridul Alam', 'faridul.alam@example.com',
'01710000010', '321 Bashundhara, Dhaka'),
SELECT * FROM dual;
```


Query Results

BENEFICIARYID	NAME	EMAIL	PHONE	ADDRESS
1	Ayesha Rahman	ayesha.rahman@example.com	01710000001	123 Green Road, Dhaka
2	Karim Uddin	karim.uddin@example.com	01710000002	456 Gulshan Avenue, Dhaka
3	Nasrin Jahan	nasrin.jahan@example.com	01710000003	789 Dhanmondi, Dhaka
4	Habib Hasan	habib.hasan@example.com	01710000004	12 Banani, Dhaka
5	Shamima Akhter	shamima.akhter@example.com	01710000005	34 Uttara, Dhaka
6	Rafiq Mia	rafiq.mia@example.com	01710000006	56 Mohammadpur, Dhaka
7	Rubina Sultana	rubina.sultana@example.com	01710000007	78 Mirpur, Dhaka
8	Biplob Hossain	biplob.hossain@example.com	01710000008	90 Tejgaon, Dhaka
9	Salma Khatun	salma.khatun@example.com	01710000009	102 Badda, Dhaka
10	Faridul Alam	faridul.alam@example.com	01710000010	321 Bashundhara, Dhaka

Distribution

```
INSERT ALL
    INTO Distribution
        VALUES (1, 1, 1, 2000.00, TO_DATE('2025-12-10', 'YYYY-MM-DD'))
    INTO Distribution
        VALUES (2, 2, 2, 1500.00, TO_DATE('2025-08-05', 'YYYY-MM-DD'))
    INTO Distribution
        VALUES (3, 3, 3, 5000.00, TO_DATE('2025-07-15', 'YYYY-MM-DD'))
    INTO Distribution
        VALUES (4, 4, 4, 3000.00, TO_DATE('2025-03-25', 'YYYY-MM-DD'))
    INTO Distribution
        VALUES (5, 5, 5, 4000.00, TO_DATE('2025-06-20', 'YYYY-MM-DD'))
    INTO Distribution
        VALUES (6, 6, 6, 1000.00, TO_DATE('2025-11-15', 'YYYY-MM-DD'))
    INTO Distribution
        VALUES (7, 7, 7, 2500.00, TO_DATE('2025-07-21', 'YYYY-MM-DD'))
    INTO Distribution
        VALUES (8, 8, 8, 8000.00, TO_DATE('2025-05-10', 'YYYY-MM-DD'))
    INTO Distribution
        VALUES (9, 9, 9, 6000.00, TO_DATE('2025-01-15', 'YYYY-MM-DD'))
    INTO Distribution
        VALUES (10, 10, 10, 3500.00, TO_DATE('2025-02-20', 'YYYY-MM-DD'))
SELECT * FROM dual;
```



Query Results

DISTRIBUTIONID	DONATIONID	BENEFICIARYID	AMOUNTGIVEN	DATEGIVEN
1	1	1	2000	10-DEC-25
2	2	2	1500	05-AUG-25
3	3	3	5000	15-JUL-25
4	4	4	3000	25-MAR-25
5	5	5	4000	20-JUN-25
6	6	6	1000	15-NOV-25
7	7	7	2500	21-JUL-25
8	8	8	8000	10-MAY-25
9	9	9	6000	15-JAN-25
10	10	10	3500	20-FEB-25

← Back

9. User Interface Design

The Donation Management System features a **web-based user interface** built using **HTML, CSS, JavaScript, and PHP**, and it runs on a local server powered by **XAMPP**. The interface is designed to be user-friendly, intuitive, and responsive, allowing administrators to manage donations, campaigns, beneficiaries, and users efficiently.

Interface Components:

1. **Home Dashboard**
   The homepage displays a navigation bar and system overview. Users can access main modules such as Donors, Donations, Beneficiaries, Campaigns, and Reports. It provides a clean and professional layout using CSS styling and Bootstrap for responsiveness.

2. **Donor Registration Form**
   This form allows the admin to add new donors by filling in details such as name, email, phone number, and address. Upon submission, the data is sent via PHP to the backend and stored in the database. Input validation is handled with JavaScript.

3. **Donation Entry Page**
   Users can enter new donations by selecting an existing donor and entering donation details like amount, date, payment method, and purpose. The system uses dropdowns, date pickers, and form validation for a smooth user experience.

4. **Beneficiary & Campaign Management**
   Separate pages are available for managing beneficiaries and campaigns. Forms include fields for contact information, need descriptions, campaign start/end dates, and target amounts. The interface ensures clarity and prevents data entry errors.

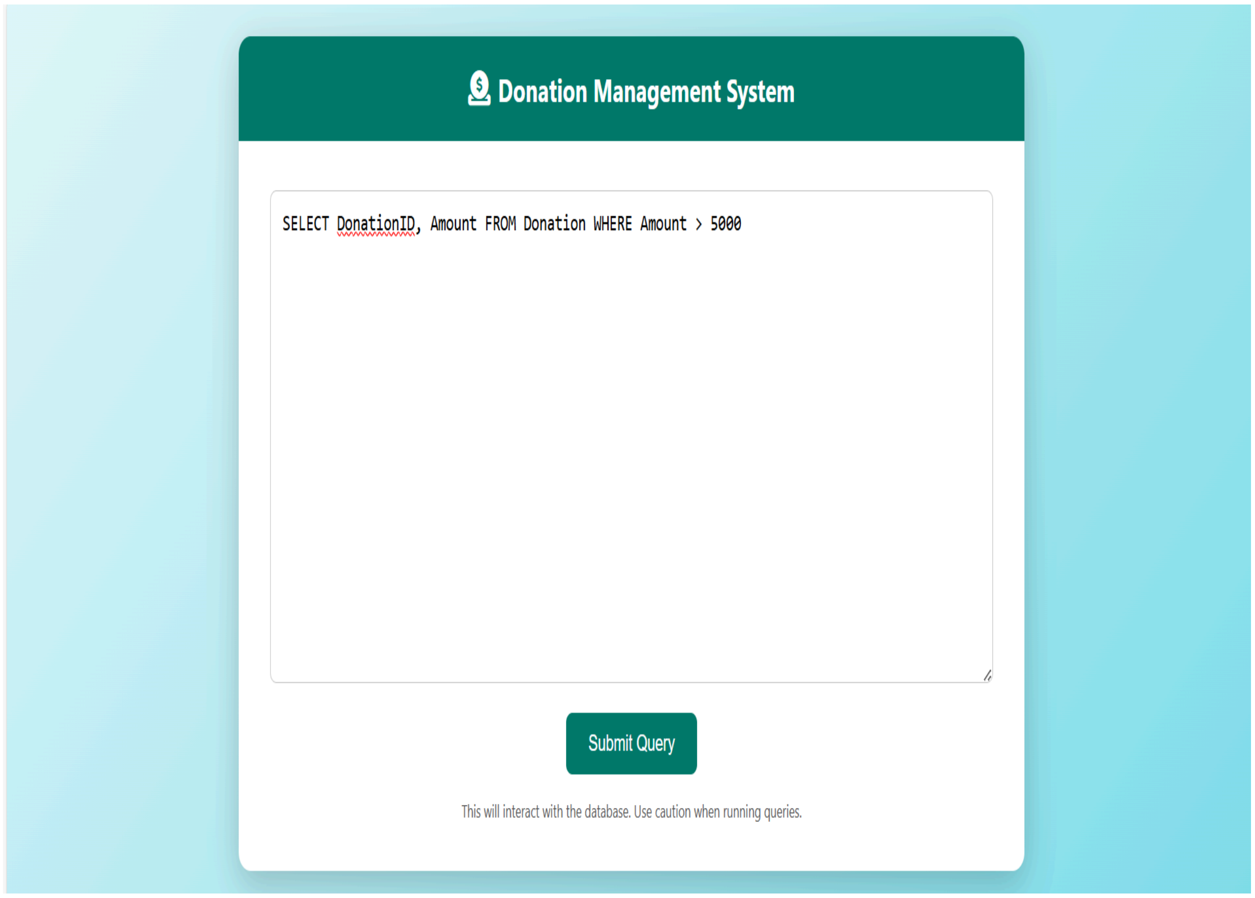
5. **Distribution Page**
   This section links donations to beneficiaries through a clean form that includes selection lists for donation IDs, beneficiary IDs, campaign involvement, and admin responsibility.

6. **Reports & Tables**
   A report section presents data from various tables in a readable format using PHP and MySQL queries. Data tables are styled with CSS and support sorting, filtering, and pagination using JavaScript or libraries like DataTables.

Technologies Used:

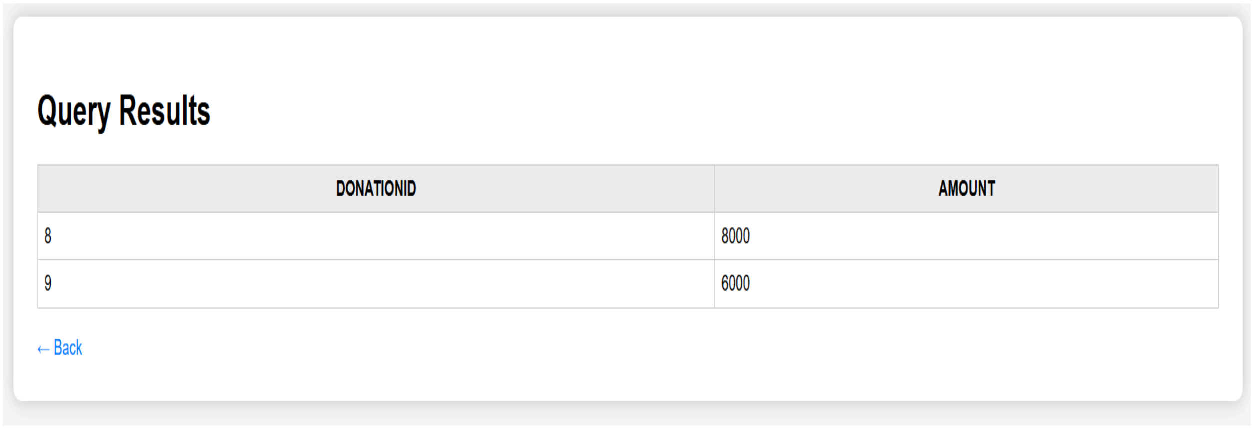
- **Frontend:** HTML5, CSS3, JavaScript (with optional Bootstrap for styling)
- **Backend:** PHP
- **Database:** MySQL (managed via phpMyAdmin)
- **Server Environment:** XAMPP (Apache & MySQL)

This user interface provides a complete and efficient platform for managing the donation workflow, offering real-time interaction with the database and ensuring data consistency through server-side processing.



```
SELECT DonationID, Amount FROM Donation WHERE Amount > 5000
```

This will interact with the database. Use caution when running queries.



Query Results

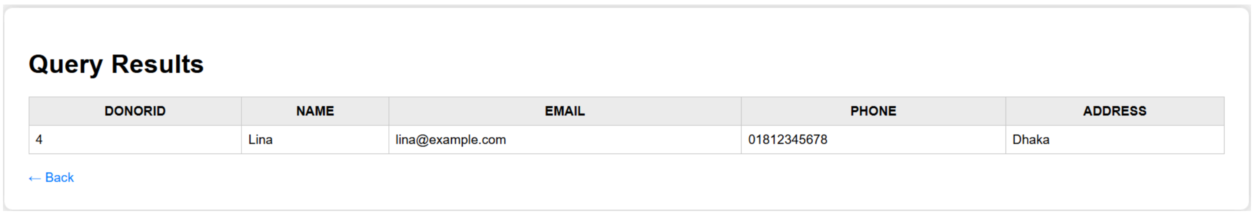
DONATIONID	AMOUNT
8	8000
9	6000

Figure 2: Text-Based User Interface for Donation Management System

10. DML Queries

10.1 Single Table Queries :

1.SELECT * FROM Donor WHERE Address LIKE '%Dhaka%';

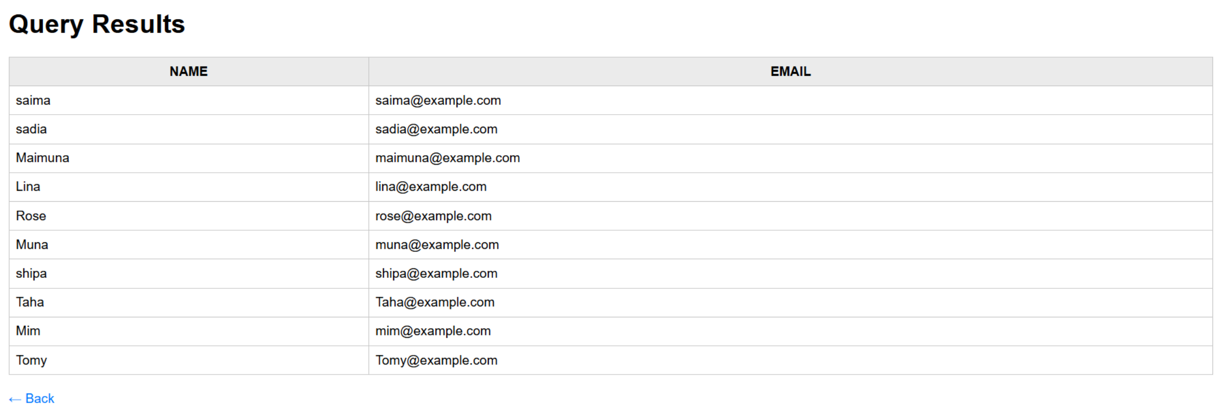


Query Results

DONORID	NAME	EMAIL	PHONE	ADDRESS
4	Lina	lina@example.com	01812345678	Dhaka

← Back

2.SELECT Name, Email FROM Donor;

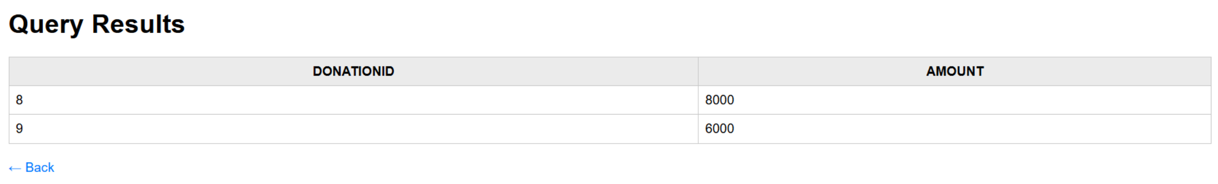


Query Results

NAME	EMAIL
saima	saima@example.com
sadia	sadia@example.com
Maimuna	maimuna@example.com
Lina	lina@example.com
Rose	rose@example.com
Muna	muna@example.com
shipa	shipa@example.com
Taha	Taha@example.com
Mim	mim@example.com
Tomy	Tomy@example.com

← Back

3.SELECT DonationID, Amount FROM Donation WHERE Amount > 5000;

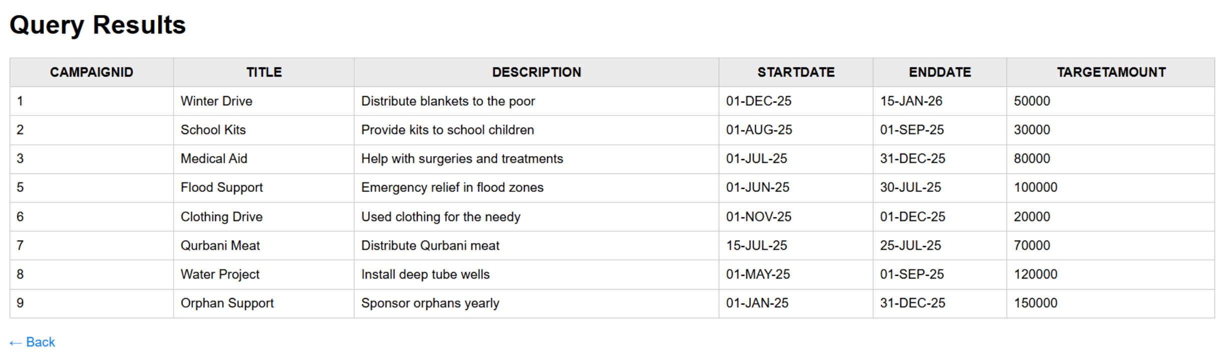


Query Results

DONATIONID	AMOUNT
8	8000
9	6000

← Back

4.SELECT * FROM Campaign WHERE EndDate > CURRENT_DATE

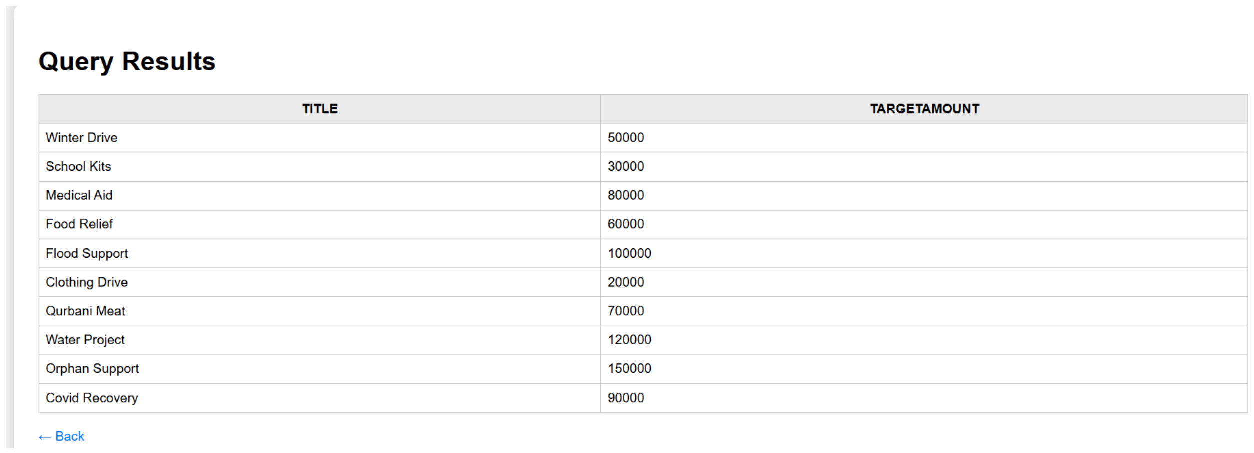


Query Results

CAMPAIGNID	TITLE	DESCRIPTION	STARTDATE	ENDDATE	TARGETAMOUNT
1	Winter Drive	Distribute blankets to the poor	01-DEC-25	15-JAN-26	50000
2	School Kits	Provide kits to school children	01-AUG-25	01-SEP-25	30000
3	Medical Aid	Help with surgeries and treatments	01-JUL-25	31-DEC-25	80000
5	Flood Support	Emergency relief in flood zones	01-JUN-25	30-JUL-25	100000
6	Clothing Drive	Used clothing for the needy	01-NOV-25	01-DEC-25	20000
7	Qurbani Meat	Distribute Qurbani meat	15-JUL-25	25-JUL-25	70000
8	Water Project	Install deep tube wells	01-MAY-25	01-SEP-25	120000
9	Orphan Support	Sponsor orphans yearly	01-JAN-25	31-DEC-25	150000

← Back

5.SELECT Title, TargetAmount FROM Campaign WHERE TargetAmount > 10000

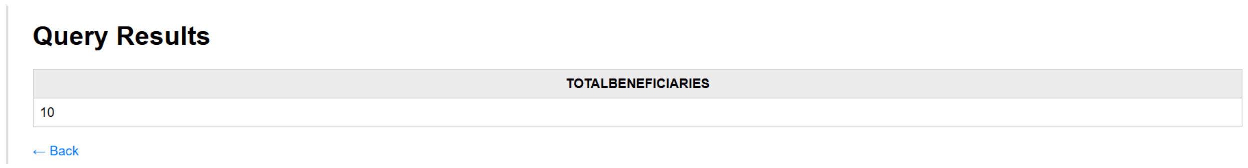


Query Results

TITLE	TARGETAMOUNT
Winter Drive	50000
School Kits	30000
Medical Aid	80000
Food Relief	60000
Flood Support	100000
Clothing Drive	20000
Qurbani Meat	70000
Water Project	120000
Orphan Support	150000
Covid Recovery	90000

← Back

6.SELECT COUNT(*) AS TotalBeneficiaries FROM Beneficiary

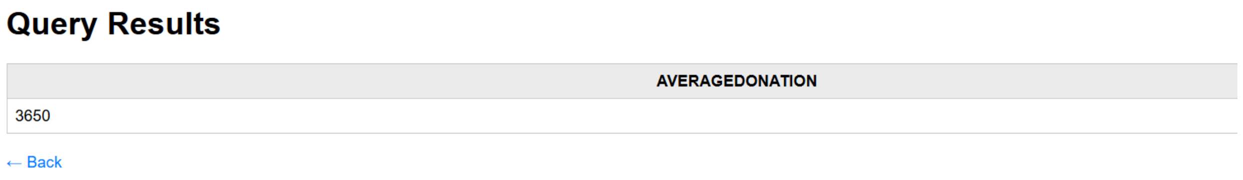


Query Results

TOTALBENEFICIARIES
10

← Back

7.SELECT AVG(Amount) AS AverageDonation FROM Donation

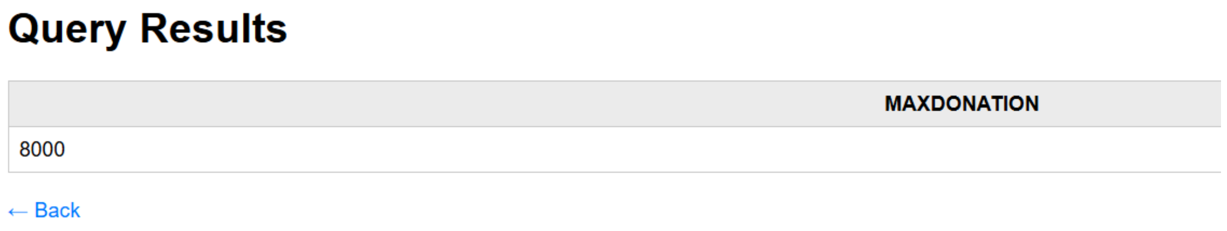


Query Results

AVERAGEDONATION
3650

← Back

8.SELECT MAX(Amount) AS MaxDonation FROM Donation

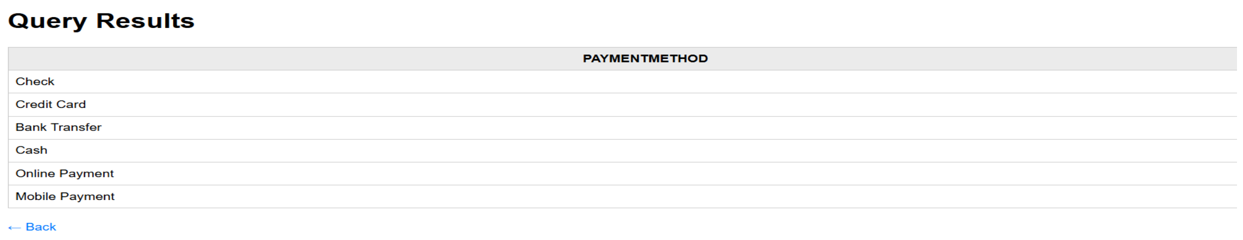


Query Results

MAXDONATION
8000

← Back

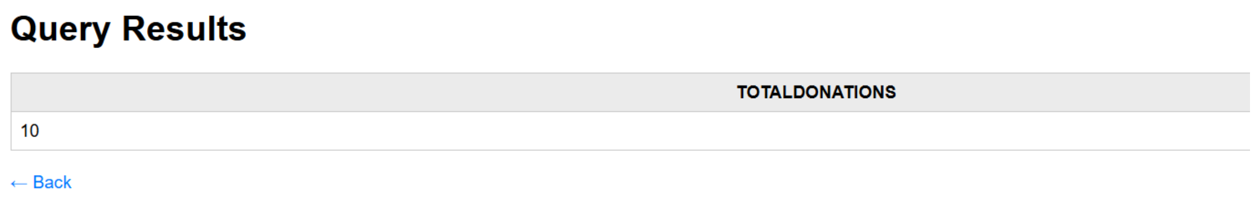
9.SELECT DISTINCT PaymentMethod FROM Donation


Query Results

PAYMENTMETHOD
Check
Credit Card
Bank Transfer
Cash
Online Payment
Mobile Payment

← Back

10.SELECT COUNT(*) AS TotalDonations FROM Donation

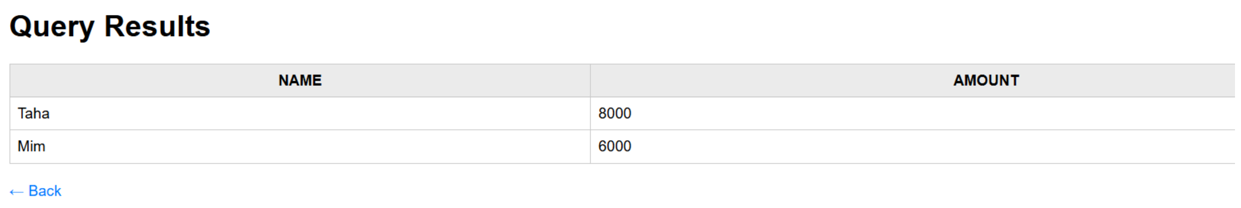

Query Results

TOTALDONATIONS
10

← Back

10.2 Multi-Table Queries :

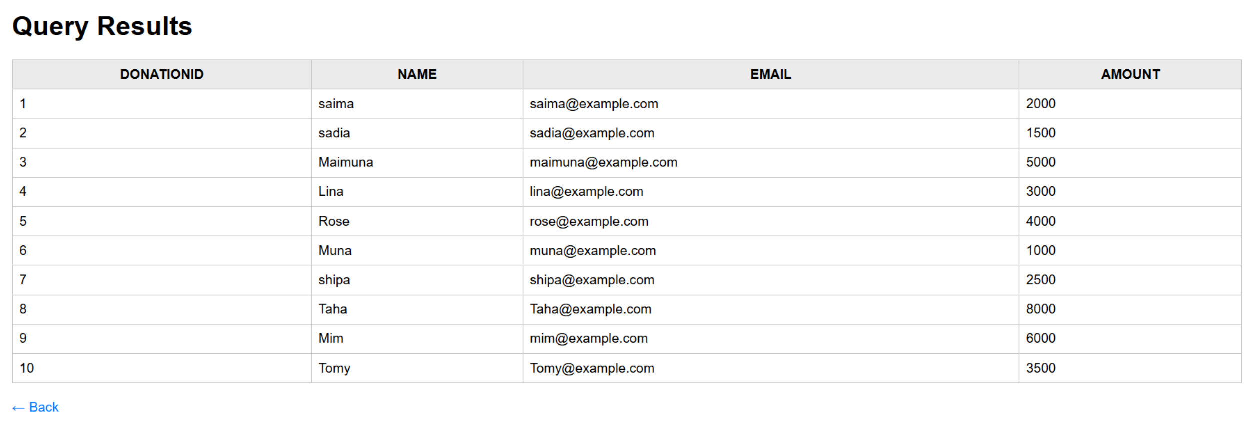
1.SELECT r.Name, d.Amount
FROM Donor r
JOIN Donation d ON r.DonorID = d.DonorID
WHERE d.Amount > 5000


Query Results

NAME	AMOUNT
Taha	8000
Mim	6000

← Back

2.SELECT d.DonationID, r.Name, r.Email, d.Amount
FROM Donation d
JOIN Donor r ON d.DonorID = r.DonorID

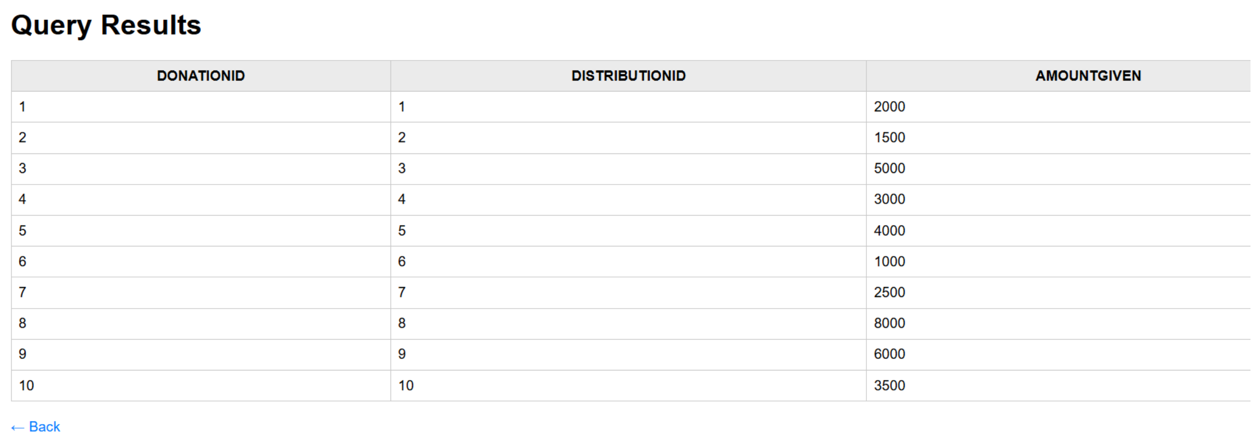


Query Results

DONATIONID	NAME	EMAIL	AMOUNT
1	saima	saima@example.com	2000
2	sadia	sadia@example.com	1500
3	Maimuna	maimuna@example.com	5000
4	Lina	lina@example.com	3000
5	Rose	rose@example.com	4000
6	Muna	muna@example.com	1000
7	shipa	shipa@example.com	2500
8	Taha	Taha@example.com	8000
9	Mim	mim@example.com	6000
10	Tomy	Tomy@example.com	3500

← Back

3.SELECT d.DonationID, dist.DistributionID, dist.AmountGiven
FROM Donation d
JOIN Distribution dist ON d.DonationID = dist.DonationID;

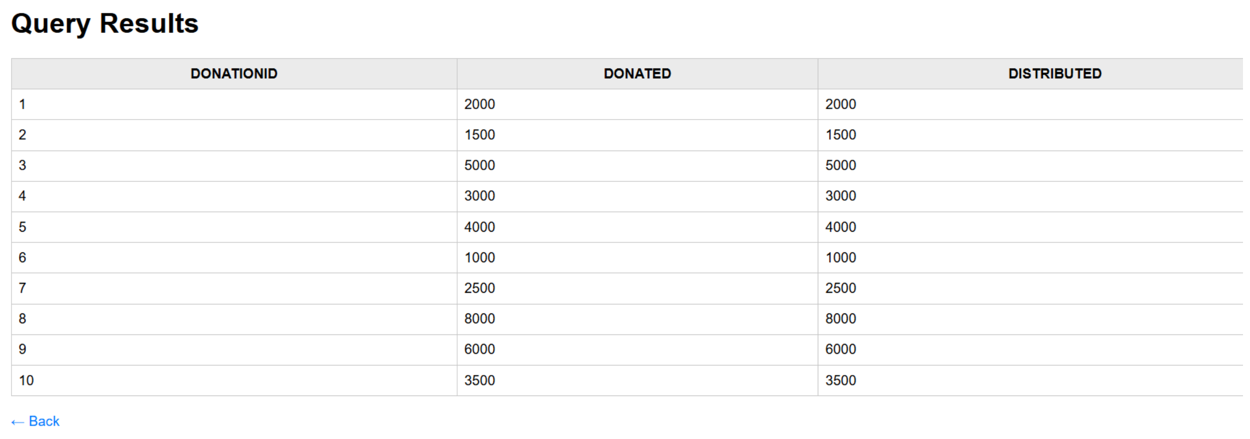


Query Results

DONATIONID	DISTRIBUTIONID	AMOUNTGIVEN
1	1	2000
2	2	1500
3	3	5000
4	4	3000
5	5	4000
6	6	1000
7	7	2500
8	8	8000
9	9	6000
10	10	3500

← Back

4.SELECT d.DonationID, d.Amount AS Donated, dist.AmountGiven AS Distributed
FROM Donation d
JOIN Distribution dist ON d.DonationID = dist.DonationID;

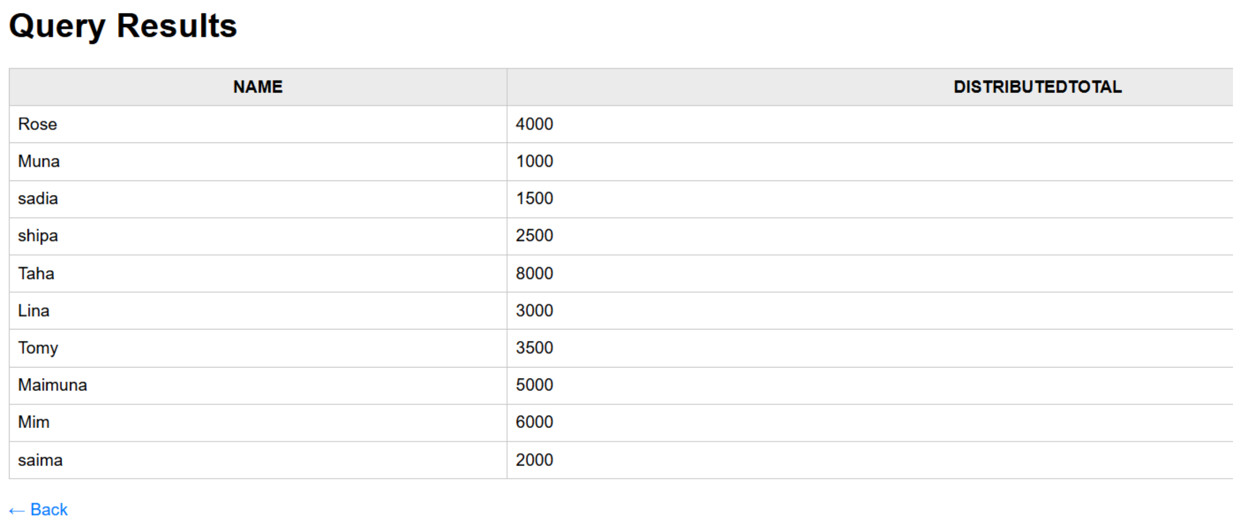


Query Results

DONATIONID	DONATED	DISTRIBUTED
1	2000	2000
2	1500	1500
3	5000	5000
4	3000	3000
5	4000	4000
6	1000	1000
7	2500	2500
8	8000	8000
9	6000	6000
10	3500	3500

← Back

5.SELECT r.Name, SUM(dist.AmountGiven) AS DistributedTotal
FROM Donor r
JOIN Donation d ON r.DonorID = d.DonorID
JOIN Distribution dist ON d.DonationID = dist.DonationID
GROUP BY r.Name;



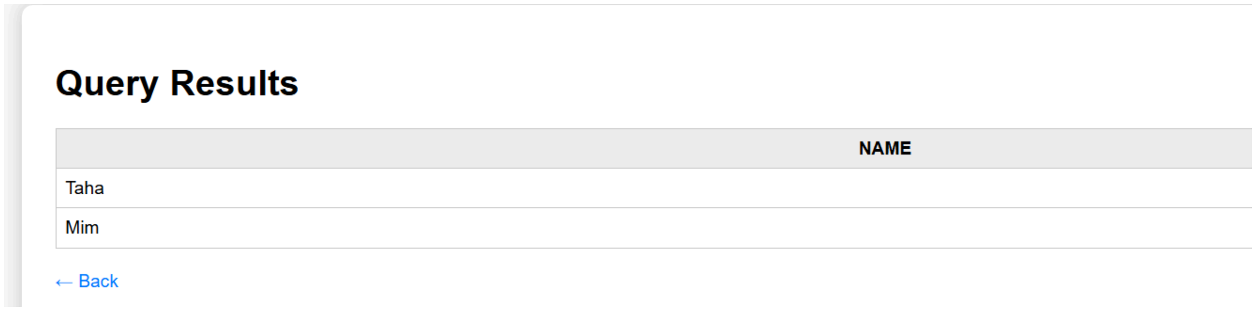
Query Results

NAME	DISTRIBUTEDTOTAL
Rose	4000
Muna	1000
sadia	1500
shipa	2500
Taha	8000
Lina	3000
Tomy	3500
Maimuna	5000
Mim	6000
saima	2000

← Back

10.3 Subqueries

1.SELECT Name FROM Donor
WHERE DonorID IN (
    SELECT DonorID FROM Donation WHERE Amount > 5000
);

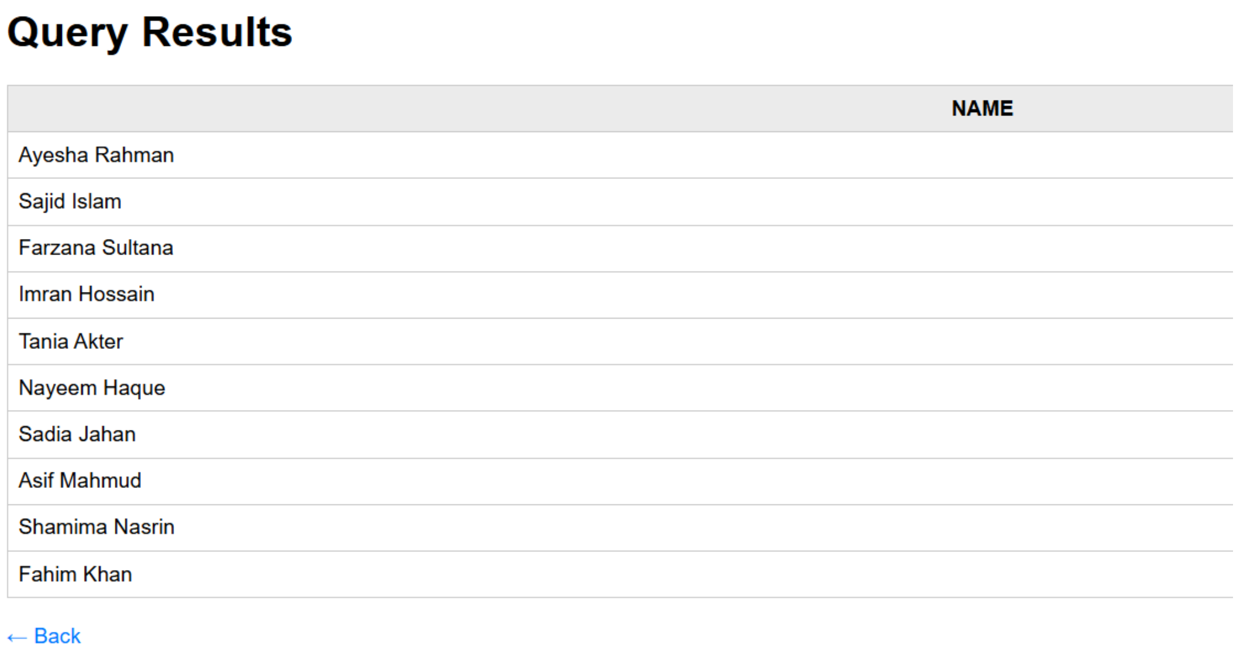


Query Results

NAME
Taha
Mim

← Back

2.SELECT Name FROM Admin
WHERE AdminID IN (
    SELECT AdminID FROM Distribution
);

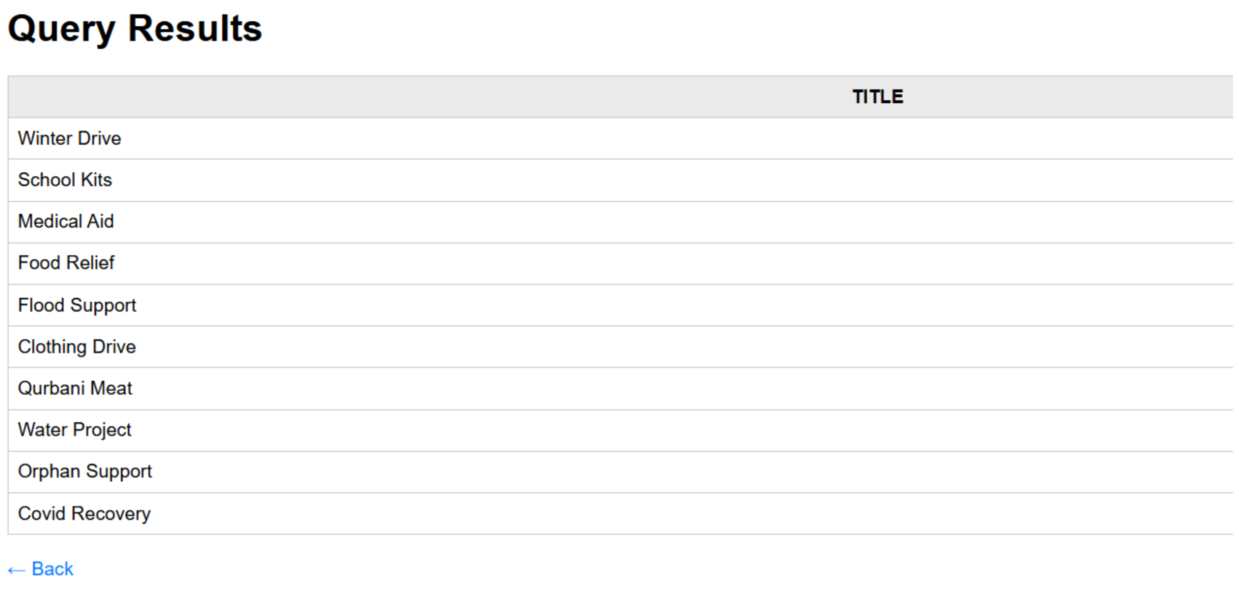


Query Results

NAME
Ayesha Rahman
Sajid Islam
Farzana Sultana
Imran Hossain
Tania Akter
Nayeem Haque
Sadia Jahan
Asif Mahmud
Shamima Nasrin
Fahim Khan

← Back

```
3.SELECT Title FROM Campaign
WHERE CampaignID IN (
    SELECT CampaignID FROM Distribution
    WHERE AmountGiven = ANY (
        SELECT Amount FROM Donation
    )
);
```

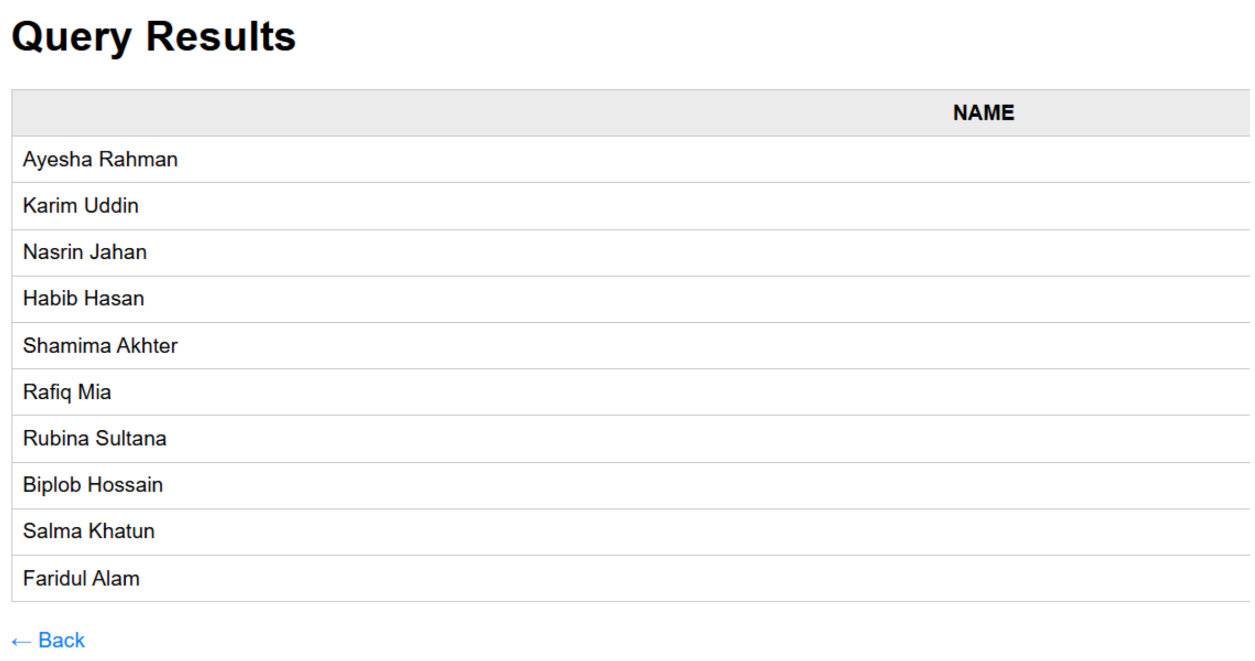


Query Results

TITLE
Winter Drive
School Kits
Medical Aid
Food Relief
Flood Support
Clothing Drive
Qurbani Meat
Water Project
Orphan Support
Covid Recovery

← Back

```
4.SELECT Name FROM Beneficiary
WHERE BeneficiaryID IN (
    SELECT BeneficiaryID FROM Distribution
);
```

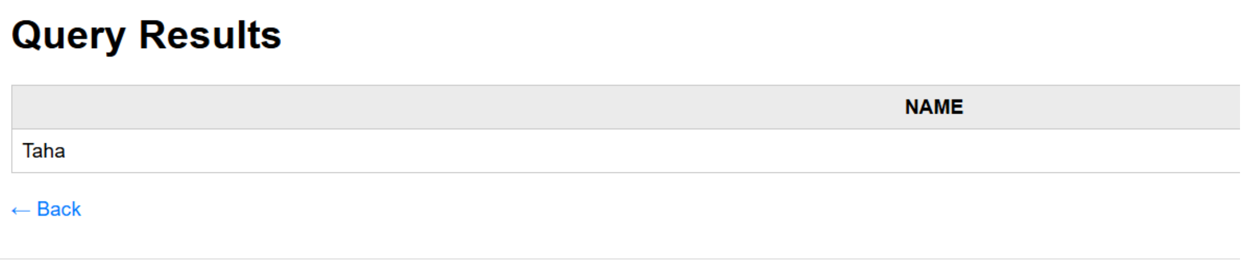


Query Results

NAME
Ayesha Rahman
Karim Uddin
Nasrin Jahan
Habib Hasan
Shamima Akhter
Rafiq Mia
Rubina Sultana
Biplob Hossain
Salma Khatun
Faridul Alam

← Back

5.
```
SELECT Name FROM Donor
WHERE DonorID IN (
    SELECT DonorID FROM Donation
    WHERE Amount = ALL (
        SELECT MAX(Amount) FROM Donation
    )
);
```



Query Results

NAME
Taha

← Back

11. Challenges Faced

During the development of the Donation Management System, we encountered several technical and structural challenges. These were addressed through debugging, research, and tool configuration.

1. Foreign Key Constraint Errors

Initially, foreign key constraint violations occurred when inserting data into tables like `Donation` and `Distribution` without ensuring the related records (e.g., Donors, Beneficiaries) existed. The solution was to insert parent table data first and use consistent IDs to maintain referential integrity.

2. Normalization Difficulties

Transforming data from unnormalized form to 3NF involved identifying and eliminating redundant or transitive dependencies. For example, moving `AdminName` and `CampaignTitle` into separate, normalized tables reduced duplication and improved design clarity.

3. CLOB Data Handling

Oracle's `CLOB` data type (used for campaign descriptions) posed challenges in terms of insertion and display, especially when interfacing with front-end forms. We resolved this by using PHP's `oci_bind_by_name()` and properly escaping special characters when submitting CLOB data.

4. PHP to Oracle Database Connection Issues

Establishing a stable PHP-to-Oracle connection was one of the most significant challenges. Problems included:

- Missing Oracle client libraries (like `OCI8`) on the server.
- Connection authentication errors due to incorrect hostname, service name, or credentials.
- Compatibility issues between XAMPP and Oracle drivers.

Solution:

- Installed and configured the OCI8 PHP extension.
- Enabled proper user privileges on Oracle (e.g., `GRANT CONNECT, RESOURCE TO username;`).

5. Session and Data Persistence

Managing user sessions during login (especially for Admins) required session handling in PHP and linking them to the database-stored usernames securely. This was solved using PHP sessions and password checks within SQL queries.

12. Conclusion

The development of the Donation Management System provided valuable hands-on experience in database design, normalization, and implementation using Oracle. Through this project, we gained a deeper understanding of how to translate real-world requirements—such as donor management, campaign tracking, and fund distribution—into a well-structured relational database system.

Key learning outcomes include the ability to:

- Design and normalize a database schema up to the Third Normal Form (3NF).
- Use ER diagrams to map relationships between real-world entities like Donors, Beneficiaries, Campaigns, and Admins.
- Implement robust database constraints to ensure data integrity.
- Handle common challenges such as foreign key dependencies, CLOB data types, and PHP-to-Oracle connectivity.
- Integrate front-end technologies with backend Oracle databases using PHP for dynamic data operations.

This project mirrors real-life systems used by NGOs, charitable foundations, and relief organizations to manage donations efficiently. By ensuring accurate record-keeping, transparency, and secure access control, the system offers a scalable foundation that can be extended into a full-featured web or mobile application.

Overall, this project not only strengthened our technical skills in database management and server-side programming but also highlighted the importance of reliable systems in socially impactful domains.

13. References

1. A. Silberschatz, H. F. Korth, and S. Sudarshan, *Database System Concepts*, 6th ed. New York, NY, USA: McGraw-Hill, 2011.
2. Oracle Corporation, "Oracle® Database SQL Language Reference", [Online]. Available: https://docs.oracle.com/en/database/oracle/oracle-database/
3. PHP Manual, "OCI8: Oracle Database Driver for PHP", [Online]. Available: https://www.php.net/manual/en/book.oci8.php
4. TutorialsPoint, "ER Model – Basic Concepts", [Online]. Available: https://www.tutorialspoint.com/dbms/er_model_basic_concepts.htm.