

**UNIVERSITY OF RWANDA  
COLLEGE OF BUSINESS AND ECONOMICS  
SCHOOL OF BUSINESS**

**DEPARTMENT OF BUSINESS INFORMATION AND TECHNOLOGY**

**DATABASE PROJECT**

**NAME; HABIMANA NGUMIRE 222001797**

**LECTURE NAME; DR BIGINGO EMMANUEL**

**PROJECT NAME; FOOD WASTAGE REDUCTION MANAGEMENT SYSTEM**

**SECTION 1**

1. **1. Describe all the entities and their corresponding attributes that are in your database.**

Certainly, in a food wastage reduction management system project, you can identify various entities and their attributes to effectively manage and reduce food waste. Here are some key entities and their associated attributes:

1. Food Items:

- Name

- Category (e.g., fruits, vegetables, dairy)

- Expiration Date

- Quantity

- Location (where it's stored)

- Purchase Date

- Donation Status

2.Users:

user id

Food id

Report id

Name

- Contact Information

- Role

- Address

3. Donation Requests:

* Donation id
* User id
* Report id

- List

- Date Requested

- Status

4. Inventory Transactions:

Inventory id

User id

Food id

- Food Item

- Quantity In

- Quantity Out

- Date

- Source purchase

5. Notifications:

Notification id

User id

- Recipient

- Message

- Date/Time

6. Reports:

Report id

- Date Range

- Type of Report

- Parameters

7. Analytics:

Analytic id

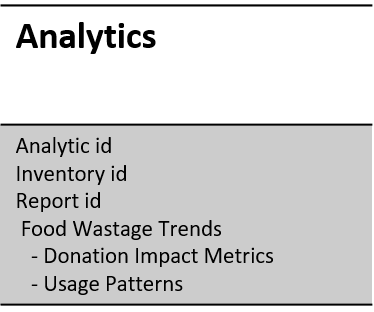
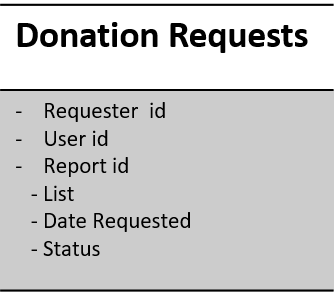
Inventory id

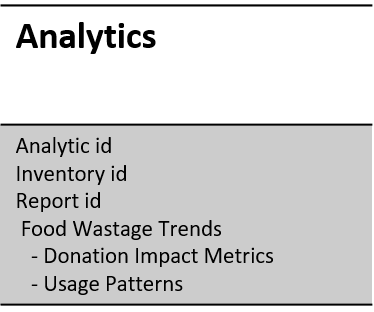
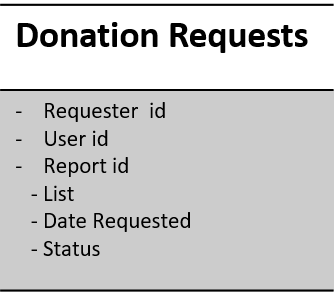
Report id

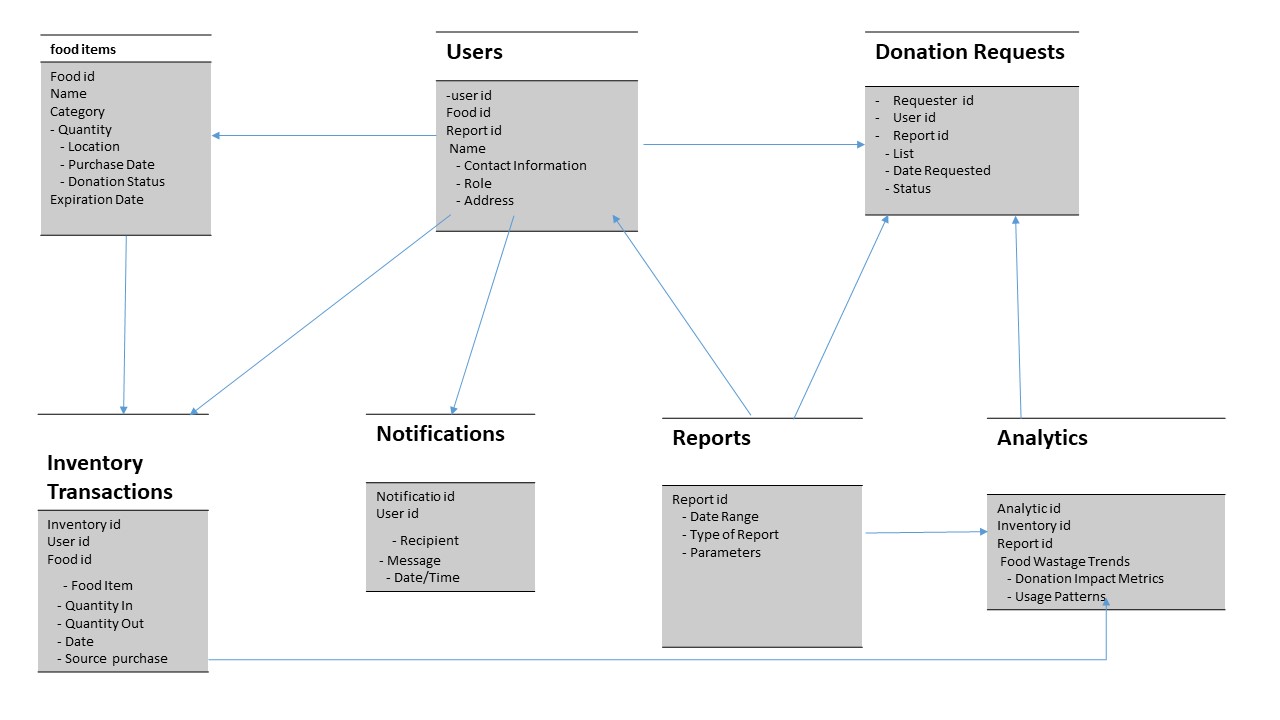
Food Wastage Trends

- Donation Impact Metrics

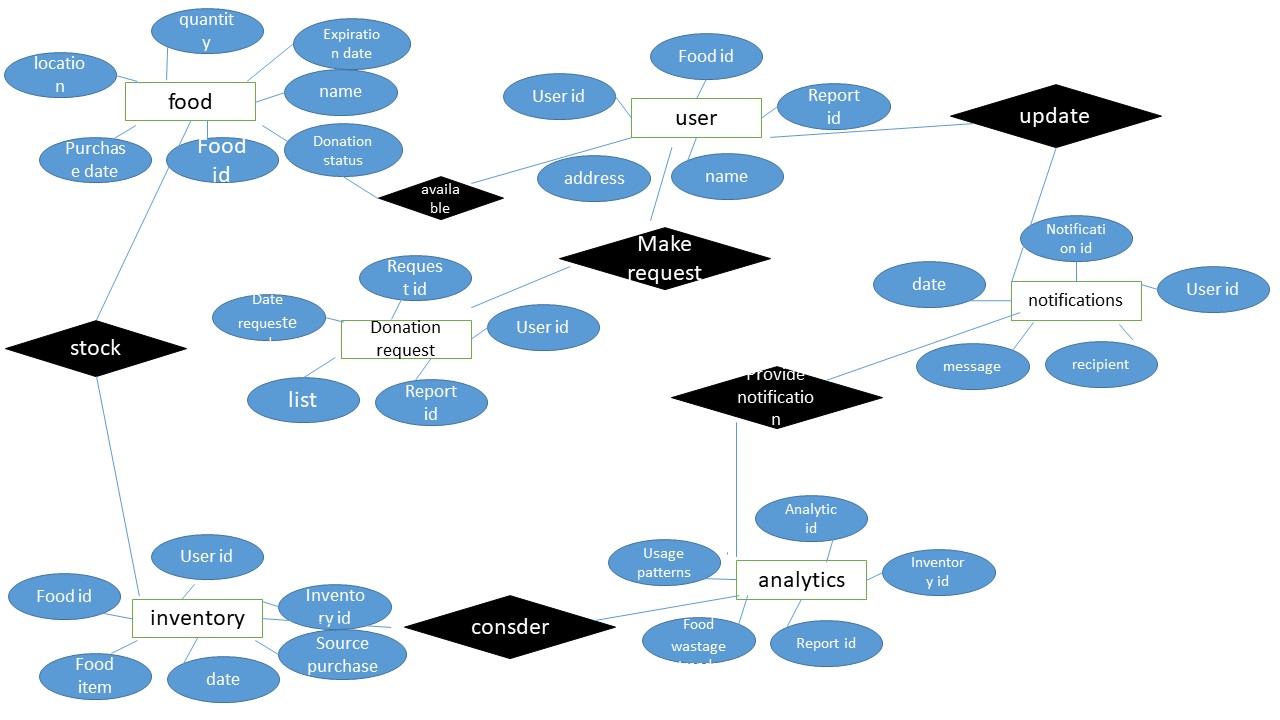
- Usage Patterns

1. **Create an LDM of your system**





**3.Create an ERD of your system**



Section ll

**1.Create the database of your system.**

mysql -u root

show databases;

create database habimana;

user ngumire;

**2.Write queries to create all the tables and relationships of your system**.

**Table of**

Create table food items(

->food \_id int auto\_increment primary key,

-> Names varchar(225) not null,

->category varchar(200) not null,

-> purchase date varchar(10)

-> ,

->donation status varchar(40) not null,

->expiration date varchar(40) not null,

-> );

Create table user (

->user \_id int auto\_increment primary key,

-> Names varchar(225) not null,

-> report\_ id,

-> food \_id varchar(10),

->phone number varchar(200) not null,

-> Address varchar(40) not null

-> );

Table of donation request

Create table donation request (

->requestor \_id int auto\_increment primary key,

->user\_id varchar(50) not null,

->request\_id varchar(255) not null,

-> list varchar100) not null,

-> date requested varchar(50) not null,

-> status varchar(50)not null

-> );

Table of inventory

Create table inventory(

-> inventory\_id int auto\_increment primary key,

-> Date date,

-> user\_id varchar(50) not null,

-> food\_id varchar(50) not null,

- quantity in int,

-> Foreign key(user\_id) references job(user\_id),

-> Foreign key(food\_id) references worker(food\_id)

-> );

Table of notifications

Create table notifications(

->notification \_id int auto\_increment primary key,

-> user\_id,

-> receptionist varchar(50) not null,

->rdate varchar(50) not null,

->message varchal(223)not null,

-> Foreign key(user\_id) references worker(user\_id)

-> );

Table of report

Create table report(

-> report\_id int auto\_increment primary key,

-> date varchar(50) not null,

-> report name varchar(50) not null,

->parameters varchar(50) not null

-> );

Tables of analysits

Create tables for analysists

->analytic\_id int auto\_increment primary key,

->report\_id varchar(225) not null,

->food wastage trends varchar(200) not null,

-> donation impact varchar(10)

-> ,inventory\_id int,

->usage platform varchar(40) not null

-> );

3.**Write queries to insert data into your tables**

INSERT INTO FoodItems (item\_name, category, expiration\_date, quantity, unit)

VALUES ('Apples', 'Fruit', '2023-10-15', 100, 'pounds');

INSERT INTO Users (username, full\_name, email, password)

VALUES ('john\_doe', 'John Doe', 'john@example.com', 'hashed\_password');

INSERT INTO DonationRequests (user\_id, request\_date, food\_item\_id, quantity, reason)

VALUES (1, '2023-09-13', 1, 50, 'Surplus from an event');

INSERT INTO InventoryTransactions (transaction\_date, food\_item\_id, quantity\_change, reason)

VALUES ('2023-09-13', 1, 50, 'Donation received');

INSERT INTO Notifications (user\_id, message, notification\_date)

VALUES (1, 'Your donation request has been approved.', '2023-09-13');

INSERT INTO Reports (report\_date, report\_type, report\_data)

VALUES ('2023-09-13', 'Inventory', 'Inventory data for the day');

INSERT INTO Analytics (analytics\_date, metric\_name, metric\_value)

VALUES ('2023-09-13', 'FoodWasteReduction', 25.5);

**4.Write queries to display all information in your tables**

Display all information from the "FoodItems" table:

SELECT \* FROM FoodItems;

Display all information from the "Users" table

SELECT \* FROM Users;

Display all information from the "DonationRequests" table

SELECT \* FROM DonationRequests;

Display all information from the "InventoryTransactions" table

SELECT \* FROM InventoryTransactions;

Display all information from the "Notifications" table

SELECT \* FROM Notifications;

Display all information from the "Reports" table

SELECT \* FROM Reports;

Display all information from the "Analytics" table

SELECT \* FROM Analytics;

**5.Write a query to update information in any of the two tables of your choice**

Update information in the "FoodItems" table

-- Update the quantity and expiration date of a specific food item (e.g., Apples)

UPDATE FoodItems

SET quantity = 120, expiration\_date = '2023-10-30'

WHERE item\_name = 'Apples';

Update information in the "Users" table

-- Update the email address of a specific user (e.g., John Doe)

UPDATE Users

SET email = 'john.doe@example.com'

WHERE username = 'john\_doe';

**Section lll**

**1.Create a view to insert data into your tables**

CREATE VIEW FoodWastageManagement AS

SELECT

'FoodItems' AS table\_name,

item\_name,

category,

expiration\_date,

quantity,

unit

FROM FoodItems

UNION ALL

SELECT

'Users' AS table\_name,

username,

full\_name,

email

FROM Users

UNION ALL

SELECT

'DonationRequests' AS table\_name,

request\_date,

food\_item\_id,

quantity,

reason

FROM DonationRequests

UNION ALL

SELECT

'InventoryTransactions' AS table\_name,

transaction\_date,

food\_item\_id,

quantity\_change,

reason

FROM InventoryTransactions

UNION ALL

SELECT

'Notifications' AS table\_name,

user\_id,

message,

notification\_date

FROM Notifications

UNION ALL

SELECT

'Reports' AS table\_name,

report\_date,

report\_type,

report\_data

FROM Reports

UNION ALL

SELECT

'Analytics' AS table\_name,

analytics\_date,

metric\_name,

metric\_value

FROM Analytics;

**2.Create a view to display all information in your tables**

CREATE VIEW FoodWastageManagementView AS

SELECT 'FoodItems' AS TableName, \* FROM FoodItems

UNION ALL

SELECT 'Users' AS TableName, \* FROM Users

UNION ALL

SELECT 'DonationRequest' AS TableName, \* FROM DonationRequest

UNION ALL

SELECT 'InventoryTransactions' AS TableName, \* FROM InventoryTransactions

UNION ALL

SELECT 'Notifications' AS TableName, \* FROM Notifications

UNION ALL

SELECT 'Reports' AS TableName, \* FROM Reports

UNION ALL

SELECT 'Analytics' AS TableName, \* FROM Analytics;

**3.Create a view to update information in any of the two tables of your system**

Updating information in the "FoodItems" table

UPDATE FoodItems

SET quantity = 120, expiration\_date = '2023-10-30'

WHERE item\_name = 'Apples';

Updating information in the "Users" table

UPDATE Users

SET email = 'john.doe@example.com'

WHERE username = 'john\_doe';

**4.Create a view to delete data in any two tables according to any simple condition of your choice**

-- Delete food items that are expired

DELETE FROM FoodItems

WHERE expiration\_date < CURDATE();

-- Delete users who have not logged in for a year

DELETE FROM Users

WHERE last\_login\_date < DATE\_SUB(CURDATE(), INTERVAL 1 YEAR);

SELECT \* FROM FoodItemsToDelete;

-- Delete expired food items

DELETE FROM FoodItems

WHERE item\_name IN (SELECT item\_name FROM FoodItemsToDelete);

-- Delete users based on a condition (for example, users with no activity)

DELETE FROM Users

WHERE last\_activity\_date < DATE\_SUB(CURDATE(), INTERVAL 1 YEAR);

**5.In your database ,create one view of your choice that consider a sub\_query**

CREATE VIEW DonorTotalDonations AS

SELECT D.donor\_id, D.donor\_name,

(SELECT SUM(quantity)

FROM Donations AS D2

WHERE D2.donor\_id = D.donor\_id) AS total\_donation\_quantity

FROM Donors AS D;

Section llll

**1.Create a stored procedure to insert data into your tables**

-- Insert into FoodItems table

INSERT INTO FoodItems (item\_name, category, expiration\_date, quantity, unit)

VALUES (p\_item\_name, p\_category, p\_expiration\_date, p\_quantity, p\_unit);

-- Insert into Users table

INSERT INTO Users (username, full\_name, email)

VALUES (p\_username, p\_full\_name, p\_email);

-- Insert into DonationRequests table

INSERT INTO DonationRequests (user\_id, request\_date, food\_item\_id, quantity, reason)

VALUES (LAST\_INSERT\_ID(), p\_request\_date, p\_food\_item\_id, p\_donation\_quantity, p\_reason);

-- Insert into InventoryTransactions table

INSERT INTO InventoryTransactions (transaction\_date, food\_item\_id, quantity\_change, reason)

VALUES (p\_transaction\_date, p\_food\_item\_id, p\_quantity\_change, p\_transaction\_reason);

-- Insert into Notifications table

INSERT INTO Notifications (user\_id, message, notification\_date)

VALUES (LAST\_INSERT\_ID(), p\_notification\_message, p\_notification\_date);

-- Insert into Reports table

INSERT INTO Reports (report\_date, report\_type, report\_data)

VALUES (p\_report\_date, p\_report\_type, p\_report\_data);

-- Insert into Analytics table

INSERT INTO Analytics (analytics\_date, metric\_name, metric\_value)

VALUES (p\_analytics\_date, p\_metric\_name, p\_metric\_value);

END //

DELIMITER ;

CALL insert data into tables();

**2.Create a stored procedure to display all information in your tables**

-- Select and display data from FoodItems table

SELECT \* FROM FoodItems;

-- Select and display data from Users table

SELECT \* FROM Users;

-- Select and display data from DonationRequests table

SELECT \* FROM DonationRequests;

-- Select and display data from InventoryTransactions table

SELECT \* FROM InventoryTransactions;

-- Select and display data from Notifications table

SELECT \* FROM Notifications;

-- Select and display data from Reports table

SELECT \* FROM Reports;

-- Select and display data from Analytics table

SELECT \* FROM Analytics;

END //

DELIMITER ;

CALL DisplayAllDataFromTables();

**3.Create a stored procedure to update information in any of the two tables of your system**

DELIMITER //

CREATE PROCEDURE UpdateFoodItemsAndUsers(

-- Update information in the FoodItems table

UPDATE FoodItems

SET quantity = p\_new\_quantity

WHERE item\_name = p\_item\_name;

-- Update information in the Users table

UPDATE Users

SET email = p\_new\_email

WHERE username = p\_username;

END //

DELIMITER ;

CALL UpdateFoodItemsAndUsers('Apples', 150, 'john\_doe', 'new\_email@example.com');

**4.Create a stored procedure to delete data in any two of your tables according to any simple condition of your choice**

DELIMITER //

CREATE PROCEDURE DeleteExpiredFoodItemsAndInactiveUsers()

BEGIN

-- Delete expired food items from the FoodItems table

DELETE FROM FoodItems

WHERE expiration\_date < CURDATE();

-- Delete inactive users from the Users table (e.g., users who haven't logged in for a year)

DELETE FROM Users

WHERE last\_login\_date < DATE\_SUB(CURDATE(), INTERVAL 1 YEAR);

END //

DELIMITER ;

CALL DeleteExpiredFoodItemsAndInactiveUsers();

**5.in your database, stored the procedure view of your choice that considers sub\_query**.

DELIMITER //

CREATE PROCEDURE DonorDonationTotals()

BEGIN

SELECT D.donor\_id, D.donor\_name,

(SELECT SUM(quantity)

FROM Donations AS D2

WHERE D2.donor\_id = D.donor\_id) AS total\_donation\_quantity

FROM Donors AS D;

END //

DELIMITER ;

Section v

**1.Create after inserting triggers for the two of your choice**

After-Insert Trigger for the "FoodItems" Table

DELIMITER //

CREATE TRIGGER FoodItemAfterInsert

AFTER INSERT ON FoodItems FOR EACH ROW

BEGIN

-- Add your actions here, such as logging or additional operations

INSERT INTO AuditLog (table\_name, action, timestamp)

VALUES ('FoodItems', 'INSERT', NOW());

END;

//

DELIMITER ;

After-Insert Trigger for the "Users" Table

DELIMITER //

CREATE TRIGGER UserAfterInsert

AFTER INSERT ON Users FOR EACH ROW

BEGIN

-- Add your actions here, such as sending a welcome email or additional operations

INSERT INTO UserActivityLog (user\_id, activity, timestamp)

VALUES (NEW.user\_id, 'User registered', NOW());

END;

//

DELIMITER ;

**2.Create after update triggers for any two tables of your choice**

After-Update Trigger for the "FoodItems" Table

DELIMITER //

CREATE TRIGGER FoodItemAfterUpdate

AFTER UPDATE ON FoodItems FOR EACH ROW

BEGIN

-- Add your actions here, such as logging or additional operations

INSERT INTO AuditLog (table\_name, action, timestamp)

VALUES ('FoodItems', 'UPDATE', NOW());

END;

//

DELIMITER ;

After-Update Trigger for the "Users" Table

DELIMITER //

CREATE TRIGGER UserAfterUpdate

AFTER UPDATE ON Users FOR EACH ROW

BEGIN

-- Add your actions here, such as sending notifications or additional operations

INSERT INTO UserActivityLog (user\_id, activity, timestamp)

VALUES (NEW.user\_id, 'User profile updated', NOW());

END;

//

DELIMITER ;

3**.Create after delete trigger of two tables of your choice**

After-Delete Trigger for the "FoodItems" Table

DELIMITER //

CREATE TRIGGER FoodItemAfterDelete

AFTER DELETE ON FoodItems FOR EACH ROW

BEGIN

-- Add your actions here, such as logging or additional operations

INSERT INTO AuditLog (table\_name, action, timestamp)

VALUES ('FoodItems', 'DELETE', NOW());

END;

//

DELIMITER ;

After-Delete Trigger for the "Users" Table

DELIMITER //

CREATE TRIGGER UserAfterDelete

AFTER DELETE ON Users FOR EACH ROW

BEGIN

-- Add your actions here, such as sending notifications or additional operations

INSERT INTO UserActivityLog (user\_id, activity, timestamp)

VALUES (OLD.user\_id, 'User account deleted', NOW());

END;

//

DELIMITER ;

**Section vi**

**1.Create a user with your name as username and your student number as your password and grant all privileges to the created user**

mysql -u root -p

CREATE USER 'habimana'@'localhost' IDENTIFIED BY '222001797';

GRANT ALL PRIVILEGES ON \*.\* TO 'habimana'@'localhost';

FLUSH PRIVILEGES;

EXIT;

**2.Create a user with your names\_semi as username and your student number as password and give him insert ,update and delete privileges to the created user**.

CREATE USER 'habimana\_ngumire\_semi'@'localhost' IDENTIFIED BY '222001797';

GRANT INSERT, UPDATE, DELETE ON database\_name.table\_name TO 'habimana\_ngumire\_semi'@'localhost';

FLUSH PRIVILEGES;

exit;

**3.Revoke insert privileges to the last user you created.**

mysql -u root -p

REVOKE INSERT ON \*.\* FROM 'habimana ngumire\_semi'@'localhost';

FLUSH PRIVILEGES;

exit;

**END OF MY PROJECT PREPARED BY HABIMANA NGUMIRE**