**Name:** Anushka H. Nevgi

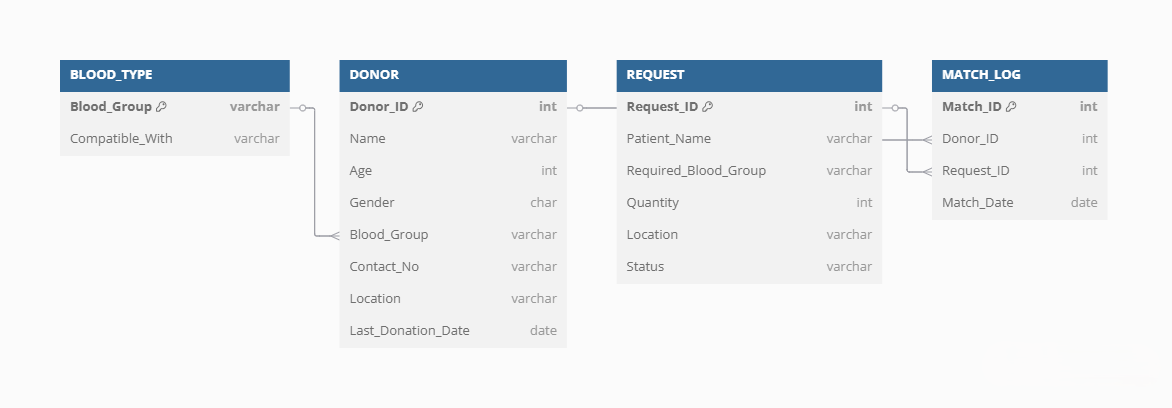
**Roll number: A**31

**Course:** Advanced Database Management System1

**Guided by:** Prof. Mrunal A. Deshpande

**Problem Statement:** In emergency situations, finding a blood donor quickly can be a matter of life and death. This project aims to maintain a structured, accessible database of registered donors, blood requests, and a matching mechanism that enables hospitals or individuals to efficiently find available blood donors based on location, blood group, and last donation date. The project is implemented using PL/SQL with procedures, triggers, and reporting queries to simulate real-time tracking and allocation.

**ER diagram:**



**Entity-Relationship Description:**

DONOR (Donor\_ID, Name, Age, Gender, Blood\_Group, Contact\_No, Location, Last\_Donation\_Date)  
BLOOD\_TYPE (Blood\_Group, Compatible\_With)  
REQUEST (Request\_ID, Patient\_Name, Required\_Blood\_Group, Quantity, Location, Status)  
MATCH\_LOG (Match\_ID, Donor\_ID, Request\_ID, Match\_Date)

**Creating tables:**

CREATE TABLE DONOR (

    Donor\_ID NUMBER PRIMARY KEY,

    Name VARCHAR2(50),

    Age NUMBER,

    Gender VARCHAR2(10),

    Blood\_Group VARCHAR2(5),

    Contact\_No VARCHAR2(15),

    Location VARCHAR2(50),

    Last\_Donation\_Date DATE

);



CREATE TABLE BLOOD\_TYPE (

    Blood\_Group VARCHAR2(5),

    Compatible\_With VARCHAR2(5)

);



CREATE TABLE REQUEST (

    Request\_ID NUMBER PRIMARY KEY,

    Patient\_Name VARCHAR2(50),

    Required\_Blood\_Group VARCHAR2(5),

    Quantity NUMBER,

    Location VARCHAR2(50),

    Status VARCHAR2(20)

);



CREATE TABLE MATCH\_LOG (

    Match\_ID NUMBER PRIMARY KEY,

    Donor\_ID NUMBER,

    Request\_ID NUMBER,

    Match\_Date DATE,

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

    FOREIGN KEY (Request\_ID) REFERENCES REQUEST(Request\_ID)

);

**Inserting data into the tables:**

-- Sample Donors

INSERT INTO DONOR VALUES (1, 'Rahul', 25, 'Male', 'A+', '9999999999', 'Pune', TO\_DATE('2024-12-01','YYYY-MM-DD'));

INSERT INTO DONOR VALUES (2, 'Sneha', 30, 'Female', 'O-', '8888888888', 'Pune', TO\_DATE('2025-03-01','YYYY-MM-DD'));

-- Blood Type Compatibility

INSERT INTO BLOOD\_TYPE VALUES ('A+', 'A+');

INSERT INTO BLOOD\_TYPE VALUES ('A+', 'O+');

INSERT INTO BLOOD\_TYPE VALUES ('O-', 'O-');

-- Sample Request

INSERT INTO REQUEST VALUES (101, 'Ravi Patil', 'A+', 1, 'Pune', 'Pending');



**Creating various procedures and implementing them:**

1. **Donor Details Update**:

CREATE OR REPLACE PROCEDURE Update\_Donor\_Details(p\_donor\_id NUMBER, p\_new\_location VARCHAR2, p\_new\_last\_donation\_date DATE) AS

BEGIN

-- Update the donor's location and last donation date

UPDATE DONOR

SET Location = p\_new\_location,

Last\_Donation\_Date = p\_new\_last\_donation\_date

WHERE Donor\_ID = p\_donor\_id;

-- Commit the changes

COMMIT;

-- Print a success message

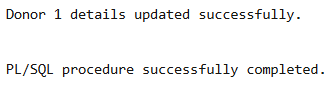
DBMS\_OUTPUT.PUT\_LINE('Donor ' || p\_donor\_id || ' details updated successfully.');

END;

/



EXEC Update\_Donor\_Details(1, 'Mumbai', TO\_DATE('2025-05-01', 'YYYY-MM-DD'));



2. **Checking Available Donors for a Specific Blood Group**:

CREATE OR REPLACE PROCEDURE List\_Available\_Donors(p\_blood\_group VARCHAR2, p\_location VARCHAR2) AS

BEGIN

    FOR donor IN (

        SELECT Donor\_ID, Name, Blood\_Group, Location, Last\_Donation\_Date

        FROM DONOR

        WHERE Blood\_Group = p\_blood\_group

          AND Location = p\_location

          AND Last\_Donation\_Date <= SYSDATE - 90  -- Donors who haven't donated in the last 90 days

    ) LOOP

        -- Display donor details

        DBMS\_OUTPUT.PUT\_LINE('Donor ID: ' || donor.Donor\_ID || ' Name: ' || donor.Name || ' Blood Group: ' || donor.Blood\_Group || ' Location: ' || donor.Location || ' Last Donation: ' || donor.Last\_Donation\_Date);

    END LOOP;

END;

/



EXEC List\_Available\_Donors('A+', 'Pune');



3. **Check Status of a Blood Request**:

CREATE OR REPLACE PROCEDURE Check\_Request\_Status(p\_request\_id NUMBER) AS

    v\_status VARCHAR2(20);

BEGIN

    -- Fetch the status of the given request ID

    SELECT Status INTO v\_status

    FROM REQUEST

    WHERE Request\_ID = p\_request\_id;

    -- Display the request status

    DBMS\_OUTPUT.PUT\_LINE('Request Status for Request\_ID ' || p\_request\_id || ': ' || v\_status);

EXCEPTION

    WHEN NO\_DATA\_FOUND THEN

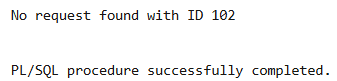
        DBMS\_OUTPUT.PUT\_LINE('No request found with ID ' || p\_request\_id);

END;

/



EXEC Check\_Request\_Status(102);



4. **Mark a Blood Request as Fulfilled**:

CREATE OR REPLACE PROCEDURE Mark\_Request\_Fulfilled(p\_request\_id NUMBER) AS

BEGIN

-- Update the request status to 'Fulfilled'

UPDATE REQUEST

SET Status = 'Fulfilled'

WHERE Request\_ID = p\_request\_id;

-- Commit the changes

COMMIT;

-- Print success message

DBMS\_OUTPUT.PUT\_LINE('Request ID ' || p\_request\_id || ' has been marked as Fulfilled.');

END;

/



EXEC Mark\_Request\_Fulfilled(102);

