**Sashiel Moonsamy**

**ST**

**Assignment 2**

**ADDB7311**

**QUESTION 1**

1. **Creating Tables**

(IIE,2024)

**--Customer Table**

CREATE TABLE Customer (

Customer\_ID NUMBER PRIMARY KEY,

First\_Name VARCHAR2(60),

Surname VARCHAR2(60),

Address VARCHAR2(110),

Contact\_Number VARCHAR2(20),

Email VARCHAR2(60)

);

**-- Employee Table**

CREATE TABLE Employee (

Employee\_ID VARCHAR2(50) PRIMARY KEY,

First\_Name VARCHAR2(60),

Surname VARCHAR2(60),

Contact\_Number VARCHAR2(20),

Address VARCHAR2(110),

Email VARCHAR2(60)

);

**-- Donator Table**

CREATE TABLE Donator (

Donator\_ID NUMBER PRIMARY KEY,

First\_Name VARCHAR2(60),

Surname VARCHAR2(60),

Contact\_Number VARCHAR2(20),

Email VARCHAR2(60)

);

(IIE,2024)

**-- Create Donation table**

CREATE TABLE Donation (

Donation\_ID NUMBER PRIMARY KEY,

Donator\_ID NUMBER REFERENCES Donator(Donator\_ID),

Donation VARCHAR2(110),

Price NUMBER(10, 3), -- Store price as a NUMBER for proper calculations

Donation\_Date DATE

);

**-- Delivery Table**

CREATE TABLE Delivery (

Delivery\_ID NUMBER PRIMARY KEY,

Delivery\_Notes VARCHAR2(210),

Dispatch\_Date DATE,

Delivery\_Date DATE

);

**-- Returns Table**

CREATE TABLE Returns (

Return\_ID VARCHAR2(15) PRIMARY KEY,

Return\_Date DATE,

Reason VARCHAR2(210),

Customer\_ID NUMBER REFERENCES Customer(Customer\_ID),

Donation\_ID NUMBER REFERENCES Donation(Donation\_ID),

Employee\_ID VARCHAR2(10) REFERENCES Employee(Employee\_ID)

);

**-- Invoice Table**

CREATE TABLE Invoice (

Invoice\_Num NUMBER PRIMARY KEY,

Customer\_ID NUMBER REFERENCES Customer(Customer\_ID),

Invoice\_Date DATE,

Employee\_ID VARCHAR2(15) REFERENCES Employee(Employee\_ID),

Donation\_ID NUMBER REFERENCES Donation(Donation\_ID),

Delivery\_ID NUMBER REFERENCES Delivery(Delivery\_ID)

);

(IIE,2024)

1. **Inserting data into Table**

**--Customer table data Inserted**

INSERT INTO Customer (Customer\_ID, First\_Name, Surname, Address, Contact\_Number, Email)

VALUES (11011, 'Jack', 'Smith', '18 Water Rd', '0877277521', 'jsmith@isat.com');

INSERT INTO Customer (Customer\_ID, First\_Name, Surname, Address, Contact\_Number, Email)

VALUES (11012, 'Pat', 'Hendricks', '22 Water Rd', '0863257857', 'ph@mcom.co.za');

INSERT INTO Customer (Customer\_ID, First\_Name, Surname, Address, Contact\_Number, Email)

VALUES (11013, 'Andre', 'Clark', '101 Summer Lane', '0834567891', 'aclark@mcom.co.za');

INSERT INTO Customer (Customer\_ID, First\_Name, Surname, Address, Contact\_Number, Email)

VALUES (11014, 'Kevin', 'Jones', '55 Mountain way', '0612547895', 'kj@isat.co.za');

INSERT INTO Customer (Customer\_ID, First\_Name, Surname, Address, Contact\_Number, Email)

VALUES (11015, 'Lucy', 'Williams', '5 Main Rd', '0827238521', 'lw@mcal.co.za');

**-- Employee table data Inserted**

INSERT INTO Employee (Employee\_ID, First\_Name, Surname, Contact\_Number, Address, Email)

VALUES ('emp101', 'Jeff', 'Davis', '0877277521', '10 Main Road', 'jand@isat.com');

INSERT INTO Employee (Employee\_ID, First\_Name, Surname, Contact\_Number, Address, Email)

VALUES ('emp102', 'Kevin', 'Marks', '0837377522', '18 Water Road', 'km@isat.com');

INSERT INTO Employee (Employee\_ID, First\_Name, Surname, Contact\_Number, Address, Email)

VALUES ('emp103', 'Adanya', 'Andrews', '0817117523', '21 Circle Lane', 'aa@isat.com');

INSERT INTO Employee (Employee\_ID, First\_Name, Surname, Contact\_Number, Address, Email)

VALUES ('emp104', 'Adebayo', 'Dryer', '0797215244', '1 Sea Road', 'aryer@isat.com');

INSERT INTO Employee (Employee\_ID, First\_Name, Surname, Contact\_Number, Address, Email)

VALUES ('emp105', 'Xolani', 'Samson', '0827122255', '12 Main Road', 'xosam@isat.com');

**-- Donator Table data Inserted**

INSERT INTO Donator (Donator\_ID, First\_Name, Surname, Contact\_Number, Email)

VALUES (20111, 'Jeff', 'Watson', '0827172250', 'jwatson@ymail.com');

INSERT INTO Donator (Donator\_ID, First\_Name, Surname, Contact\_Number, Email)

VALUES (20112, 'Stephen', 'Jones', '0837865670', 'joness@ymail.com');

INSERT INTO Donator (Donator\_ID, First\_Name, Surname, Contact\_Number, Email)

VALUES (20113, 'James', 'Joe', '0878978650', 'jj@isat.com');

INSERT INTO Donator (Donator\_ID, First\_Name, Surname, Contact\_Number, Email)

VALUES (20114, 'Kelly', 'Ross', '0826575650', 'kross@gsat.com');

INSERT INTO Donator (Donator\_ID, First\_Name, Surname, Contact\_Number, Email)

VALUES (20115, 'Abraham', 'Clark', '0797656430', 'aclark@ymail.com');

(IIE,2024)

**-- Donation table data Inserted**

INSERT INTO Donation (Donation\_ID, Donator\_ID, Donation, Price, Donation\_Date)

VALUES (7111, 20111, 'KIC Fridge', 599, TO\_DATE('01-MAY-2024', 'DD-MON-YYYY'));

INSERT INTO Donation (Donation\_ID, Donator\_ID, Donation, Price, Donation\_Date)

VALUES (7112, 20112, 'Samsung 42inch LCD', 1299, TO\_DATE('03-MAY-2024', 'DD-MON-YYYY'));

INSERT INTO Donation (Donation\_ID, Donator\_ID, Donation, Price, Donation\_Date)

VALUES (7113, 20113, 'Sharp Microwave', 1599, TO\_DATE('03-MAY-2024', 'DD-MON-YYYY'));

INSERT INTO Donation (Donation\_ID, Donator\_ID, Donation, Price, Donation\_Date)

VALUES (7114, 20115, '6 Seat Dining Room Table', 799, TO\_DATE('05-MAY-2024', 'DD-MON-YYYY'));

INSERT INTO Donation (Donation\_ID, Donator\_ID, Donation, Price, Donation\_Date)

VALUES (7115, 20114, 'Lazyboy Sofa', 1199, TO\_DATE('07-MAY-2024', 'DD-MON-YYYY'));

INSERT INTO Donation (Donation\_ID, Donator\_ID, Donation, Price, Donation\_Date)

VALUES (7116, 20113, 'JVC Surround Sound System', 179, TO\_DATE('09-MAY-2024', 'DD-MON-YYYY'));

(IIE,2024)

**-- Delivery table data Inserted**

INSERT INTO Delivery (Delivery\_ID, Delivery\_Notes, Dispatch\_Date, Delivery\_Date)

VALUES (511, 'Double packaging requested', TO\_DATE('10-MAY-2024', 'DD-MON-YYYY'), TO\_DATE('15-MAY-2024', 'DD-MON-YYYY'));

INSERT INTO Delivery (Delivery\_ID, Delivery\_Notes, Dispatch\_Date, Delivery\_Date)

VALUES (512, 'Delivery to work address', TO\_DATE('12-MAY-2024', 'DD-MON-YYYY'), TO\_DATE('15-MAY-2024', 'DD-MON-YYYY'));

INSERT INTO Delivery (Delivery\_ID, Delivery\_Notes, Dispatch\_Date, Delivery\_Date)

VALUES (513, 'Signature required', TO\_DATE('12-MAY-2024', 'DD-MON-YYYY'), TO\_DATE('17-MAY-2024', 'DD-MON-YYYY'));

INSERT INTO Delivery (Delivery\_ID, Delivery\_Notes, Dispatch\_Date, Delivery\_Date)

VALUES (514, 'No notes', TO\_DATE('12-MAY-2024', 'DD-MON-YYYY'), TO\_DATE('15-MAY-2024', 'DD-MON-YYYY'));

INSERT INTO Delivery (Delivery\_ID, Delivery\_Notes, Dispatch\_Date, Delivery\_Date)

VALUES (515, 'Birthday present wrapping required', TO\_DATE('18-MAY-2024', 'DD-MON-YYYY'), TO\_DATE('19-MAY-2024', 'DD-MON-YYYY'));

INSERT INTO Delivery (Delivery\_ID, Delivery\_Notes, Dispatch\_Date, Delivery\_Date)

VALUES (516, 'Delivery to work address', TO\_DATE('20-MAY-2024', 'DD-MON-YYYY'), TO\_DATE('25-MAY-2024', 'DD-MON-YYYY'));

-- Returns Table **data Inserted**

INSERT INTO Returns (Return\_ID, Return\_Date, Reason, Customer\_ID, Donation\_ID, Employee\_ID)

VALUES ('ret001', TO\_DATE('25-MAY-2024', 'DD-MON-YYYY'), 'Customer not satisfied with product', 11011, 7116, 'emp101');

INSERT INTO Returns (Return\_ID, Return\_Date, Reason, Customer\_ID, Donation\_ID, Employee\_ID)

VALUES ('ret002', TO\_DATE('25-MAY-2024', 'DD-MON-YYYY'), 'Product had broken section', 11013, 7114, 'emp103');

-- Invoice table **data Inserted**

INSERT INTO Invoice (Invoice\_Num, Customer\_ID, Invoice\_Date, Employee\_ID, Donation\_ID, Delivery\_ID)

VALUES (8111, 11011, TO\_DATE('15-MAY-2024', 'DD-MON-YYYY'), 'emp103', 7111, 511);

INSERT INTO Invoice (Invoice\_Num, Customer\_ID, Invoice\_Date, Employee\_ID, Donation\_ID, Delivery\_ID)

VALUES (8112, 11013, TO\_DATE('15-MAY-2024', 'DD-MON-YYYY'), 'emp101', 7114, 512);

INSERT INTO Invoice (Invoice\_Num, Customer\_ID, Invoice\_Date, Employee\_ID, Donation\_ID, Delivery\_ID)

VALUES (8113, 11012, TO\_DATE('17-MAY-2024', 'DD-MON-YYYY'), 'emp101', 7112, 513);

INSERT INTO Invoice (Invoice\_Num, Customer\_ID, Invoice\_Date, Employee\_ID, Donation\_ID, Delivery\_ID)

VALUES (8114, 11015, TO\_DATE('17-MAY-2024', 'DD-MON-YYYY'), 'emp102', 7113, 514);

INSERT INTO Invoice (Invoice\_Num, Customer\_ID, Invoice\_Date, Employee\_ID, Donation\_ID, Delivery\_ID)

VALUES (8115, 11011, TO\_DATE('17-MAY-2024', 'DD-MON-YYYY'), 'emp102', 7115, 515);

INSERT INTO Invoice (Invoice\_Num, Customer\_ID, Invoice\_Date, Employee\_ID, Donation\_ID, Delivery\_ID)

VALUES (8116, 11015, TO\_DATE('18-MAY-2024', 'DD-MON-YYYY'), 'emp103', 7116, 516);

(IIE,2024)

**QUESTION 2**

(IIE,2024)

SELECT

c.First\_Name || ', ' || c.Surname AS Customer\_Name,

-- First we need to concatenate first name and surname with a comma

i.Employee\_ID, -- This is the Employee ID from the invoice

d.Delivery\_Notes, -- This is the Delivery notes from the delivery table

don.Donation, -- This is the Donation description from the donation table

i.Invoice\_Num, -- This is the Invoice number from the invoice table

i.Invoice\_Date -- This is the Invoice date to show when filtering

FROM

Invoice i

JOIN

Customer c ON i.Customer\_ID = c.Customer\_ID -- This will join Invoice and Customer tables

JOIN

Delivery d ON i.Delivery\_ID = d.Delivery\_ID -- This will join Invoice and Delivery tables

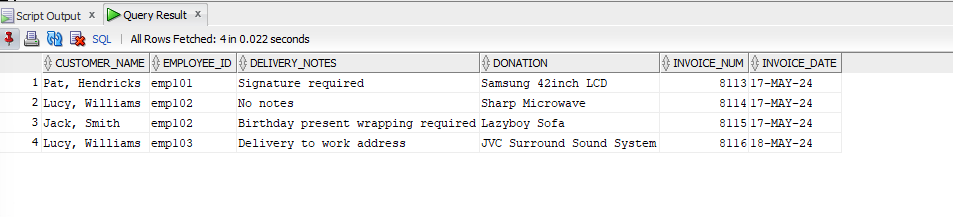
JOIN

Donation don ON i.Donation\_ID = don.Donation\_ID --This will join Invoice and Donation tables

WHERE

i.Invoice\_Date > TO\_DATE('16-MAY-2024', 'DD-MON-YYYY'); -- This will filter for invoice dates after 16 May 2024

OUTPUT



(IIE,2024)

QUESTION 3

(IIE,2024)

-- Step 1: Start by creating the Funding Table

-- Code to create the Funding table

CREATE TABLE Funding (

funding\_id NUMBER GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY, -- Now we created the auto-generating unique ID

funder VARCHAR2(110), -- This is the name of the funder

funding\_amount NUMBER (10, 3) -- This is the amount of the funding

);

-- The funding\_id column uses GENERATED BY DEFAULT AS IDENTITY to auto-generate unique IDs for each record.

-- The funder column is a VARCHAR2(100) to store the name of the funding source.

-- The funding\_amount column is a NUMBER(10, 2) to store amounts with up to 10 digits and 2 decimal places (e.g., 50000.00).

--Step 2: Example Insert Statement

-- Insert a new funding record

INSERT INTO Funding (funder, funding\_amount)

VALUES ('United Charity', 50000.00);

--The funding\_id will be automatically generated and does not need to be included in the INSERT statements.

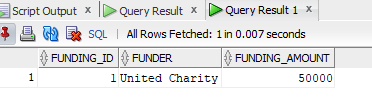
--Each of the funders and their funding amount is inserted into the table.

--Step 3: Query to View the Data

-- Query to view all the funding records

SELECT \* FROM Funding;

Output



(IIE,2024)

QUESTION 4

(IIE,2024)

SET SERVEROUTPUT ON; -- Ensure the output is enabled correctly

BEGIN

-- Cursor to fetch the required data from the Returns, Customer, and Donation tables

FOR r IN (

SELECT

c.First\_Name || ', ' || c.Surname AS Customer\_Name, -- Concatenate first name and surname

d.Donation, -- The Donation description

d.Price, -- This is the Donation price

ret.Reason -- This is the reason for the return

FROM

Returns ret

JOIN

Customer c ON ret.Customer\_ID = c.Customer\_ID -- Joins Returns with Customer

JOIN

Donation d ON ret.Donation\_ID = d.Donation\_ID -- This will join Returns with Donation

)

LOOP

-- Output the results

DBMS\_OUTPUT.PUT\_LINE('Customer: ' || r.Customer\_Name);

DBMS\_OUTPUT.PUT\_LINE('Donation Purchased: ' || r.Donation);

DBMS\_OUTPUT.PUT\_LINE('Price: ' || r.Price);

DBMS\_OUTPUT.PUT\_LINE('Return Reason: ' || r.Reason);

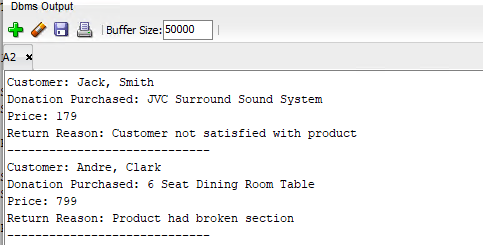
DBMS\_OUTPUT.PUT\_LINE('-----------------------------');

END LOOP;

END;

/

Output



Explanation

The “SET SERVEROUTPUT ON” command code will ensure the display of output from the PL/SQL block in the console, allowing returned donation details to be viewed(IIE,2024). The cursor provides the use of retrieving the correct data by joining the “Returns””, ‘’Customer’’, and “Donation” tables together, allowing for the right information to be extracted. The customer’s full name is formed by combining their surname and first name(IIE,2024). This will identify the individual who donated. The donation item and price of that item will be displayed to provide context, This is the reason for the return being fetched from the “Returns” table, explaining why the donation was returned. The “” DBMS\_OUTPUT.PUT\_LINE”” function prints this information in a readable format. The loop allows for each record to be processed and displayed one at a time. This setup effectively presents the reason for each donation return, along with all necessary details(IIE,2024)

.

**Question 5**

(IIE,2024)

SET SERVEROUTPUT ON;

BEGIN

-- Counter to differentiate the first customer for name formatting

DECLARE

v\_counter NUMBER := 0; -- Counter variable to track iteration

BEGIN

-- Cursor to fetch the required data from Invoice, Customer, Employee, Donation, and Delivery tables

FOR rec IN (

SELECT

c.First\_Name, -- This is the Customer First Name

c.Surname, -- This is the Customer Last Name

e.First\_Name || '.' || e.Surname AS Employee\_Name, -- Employee name with period

d.Donation, -- Donation description

TO\_CHAR(del.Dispatch\_Date, 'DD-MON-YY') AS Dispatch\_Date,

-- Formatted Dispatch Date

TO\_CHAR(del.Delivery\_Date, 'DD-MON-YY') AS Delivery\_Date,

-- Formatted Delivery Date

(del.Delivery\_Date - del.Dispatch\_Date) AS Days\_To\_Delivery

-- Calculating days between dates

FROM

Invoice i

JOIN

Customer c ON i.Customer\_ID = c.Customer\_ID

-- This will join with     Customer table

JOIN

Employee e ON i.Employee\_ID = e.Employee\_ID

-- This will join with Employee table

JOIN

Donation d ON i.Donation\_ID = d.Donation\_ID

--This will join with Donation table

JOIN

Delivery del ON i.Delivery\_ID = del.Delivery\_ID

-- This will join with Delivery table

WHERE

i.Customer\_ID = 11011

-- This will filter for customer ID 11011

)

LOOP

-- Increment the counter

v\_counter := v\_counter + 1;

-- Conditional formatting for the customer name

IF v\_counter = 1 THEN

-- For the first customer, show 'FirstName.LastName'

DBMS\_OUTPUT.PUT\_LINE('Customer: ' || rec.First\_Name || '.' || rec.Surname);

ELSE

-- For subsequent customers, show 'First Initial.LastName'

DBMS\_OUTPUT.PUT\_LINE('Customer: ' || SUBSTR(rec.First\_Name, 1, 1) || '. ' || rec.Surname);

END IF;

-- Output the remaining data

DBMS\_OUTPUT.PUT\_LINE('Employee: ' || rec.Employee\_Name);

DBMS\_OUTPUT.PUT\_LINE('Donation: ' || rec.Donation);

DBMS\_OUTPUT.PUT\_LINE('Dispatch Date: ' || rec.Dispatch\_Date);

DBMS\_OUTPUT.PUT\_LINE('Delivery Date: ' || rec.Delivery\_Date);

DBMS\_OUTPUT.PUT\_LINE('Days to Delivery: ' || rec.Days\_To\_Delivery);

DBMS\_OUTPUT.PUT\_LINE('-----------------------------------');

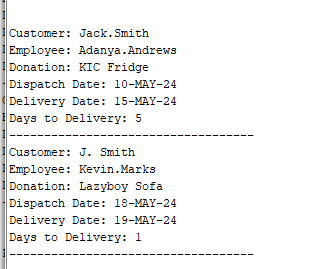
END LOOP;

END;

END;

/

OUTPUT

****

(IIE,2024)

**QUESTION 6**

(IIE,2024)

SET SERVEROUTPUT ON;

BEGIN

-- Cursor to iterate over each customer and calculate their total spending on donations

FOR rec IN (

-- Select the first name, surname, and the total amount spent by the customer

SELECT

c.First\_Name AS First\_Name, -- Retrieve the first name of the customer

c.Surname AS Surname, -- Retrieve the surname of the customer

SUM(d.Price) AS Total\_Spent -- Calculate the total amount spent by summing the donation prices

FROM

Invoice i

JOIN

Customer c ON i.Customer\_ID = c.Customer\_ID -- Join Invoice and Customer tables using Customer\_ID

JOIN

Donation d ON i.Donation\_ID = d.Donation\_ID -- Join Invoice and Donation tables using Donation\_ID

GROUP BY

c.First\_Name, c.Surname -- Group results by first name and surname to ensure totals are calculated per customer

)

LOOP

-- Output the first name of the customer

DBMS\_OUTPUT.PUT\_LINE('FIRST NAME: ' || rec.First\_Name);

-- Output the surname of the customer

DBMS\_OUTPUT.PUT\_LINE('SURNAME: ' || rec.Surname);

-- Determine the amount spent and assign a star rating if applicable

IF rec.Total\_Spent >= 1500 THEN

-- If the total amount spent is 1500 or more, display the total amount along with a 3-star rating

DBMS\_OUTPUT.PUT\_LINE('AMOUNT: R ' || rec.Total\_Spent || ' (\*\*\*)');

ELSIF rec.Total\_Spent BETWEEN 1000 AND 1499 THEN

-- If the total amount is between 1000 and 1499, display the total amount without any stars

DBMS\_OUTPUT.PUT\_LINE('AMOUNT: R ' || rec.Total\_Spent);

ELSE

-- If the total amount spent is less than 1000, display the total amount without any stars

DBMS\_OUTPUT.PUT\_LINE('AMOUNT: R ' || rec.Total\_Spent);

END IF;

-- Print a separator line to clearly distinguish between different customer records

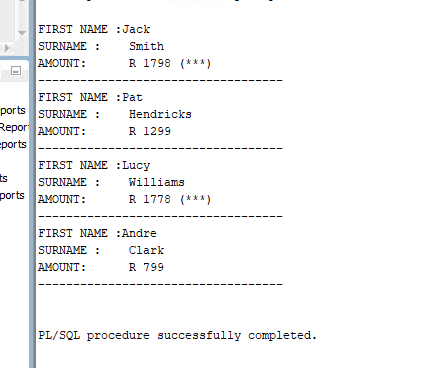
DBMS\_OUTPUT.PUT\_LINE('-----------------------------------');

END LOOP;

END;

/

**Output**

****

(IIE,2024)

**QUESTION 7**

7.1

(IIE,2024)

SET SERVEROUTPUT ON; -- Enable output to display in the console

DECLARE

-- We must declare a variable 'v\_employee\_contact' with the datatype dynamically assigned using the %TYPE attribute.

-- The datatype of 'v\_employee\_contact' will automatically match the datatype of the 'Contact\_Number' column

-- Now within the Employee table, we need to ensure theconsistency with the database structure.

v\_employee\_contact Employee.Contact\_Number%TYPE; -- %TYPE ensures that the variable inherits the exact datatype of 'Contact\_Number'

BEGIN

-- The SELECT INTO statement will be retrieving the the 'Contact\_Number' from the Employee table where Employee\_ID is 'emp102'

-- and store the value in the 'v\_employee\_contact' variable.

--The %TYPE attribute will make sure the correct variable has the appropriate datatype to hold the column value.

SELECT Contact\_Number INTO v\_employee\_contact

FROM Employee

WHERE Employee\_ID = 'emp102';

--Now displaying the contact number retrieved for the employee using DBMS\_OUTPUT happens. The %TYPE attribute

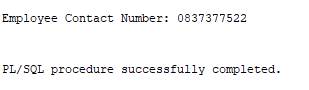
-- guarantees that the 'v\_employee\_contact' variable will be compatible with the 'Contact\_Number' column.

DBMS\_OUTPUT.PUT\_LINE ('Employee Contact Number: ' || v\_employee\_contact);

END;

/

Output



(IIE,2024)

Q7.2

(IIE,2024)

SET SERVEROUTPUT ON; -- First we need to enable the output to display in the console

DECLARE

-- Then we must declare a variable 'v\_donation' that will have the same structure as a row in the Donation table

-- Nowt the %ROWTYPE will ensure us to create a record variable that can store an entire row of data from the Donation table,

-- Now with each field corresponding to the columns in the table (e.g., Donation\_ID, Donator\_ID, Donation, Price, Donation\_Date)

v\_donation Donation%ROWTYPE; -- %ROWTYPE automation will be creating a record with all the columns of the Donation table

BEGIN

-- The SELECT INTO statement is involved in retrieving the entire row for the donation with Donation\_ID = 7112

-- Next it stores it in the 'v\_donation' variable. Since 'v\_donation' is of type %ROWTYPE, it can hold all columns from the Donation table.

SELECT \* INTO v\_donation

FROM Donation

WHERE Donation\_ID = 7112;

-- Finally the output of the details of the donation using DBMS\_OUTPUT

-- Every field of 'v\_donation' corresponds to the columns in the Donation table

DBMS\_OUTPUT.PUT\_LINE('Donation: ' || v\_donation.Donation);

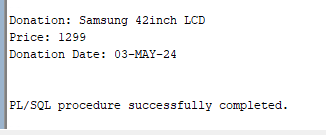
DBMS\_OUTPUT.PUT\_LINE('Price: R' || v\_donation.Price); -- The 'Price' will now be referenced as a field within 'v\_donation'

DBMS\_OUTPUT.PUT\_LINE('Donation Date: ' || v\_donation.Donation\_Date);

END;

/

OUTPUT



(IIE,2024)

Q7.3

(IIE,2024)

SET SERVEROUTPUT ON; -- Enable output to display in the console

DECLARE

-- Declaring the user-defined exception 'e\_invalid\_dates'

-- User-defined exceptions allow us to handle specific error conditions that aren't automatically raised by Oracle.

-- Here, the 'e\_invalid\_dates' is declared to handle cases where the dispatch date is after the delivery date.

e\_invalid\_dates EXCEPTION;

-- Declaring the variables to hold the dispatch and delivery dates,

-- Making %TYPE to inherit the datatype of the columns from the Delivery table

v\_dispatch\_date Delivery.Dispatch\_Date%TYPE; -- Variable to store the dispatch date

v\_delivery\_date Delivery.Delivery\_Date%TYPE; -- Variable to store the delivery date

BEGIN

-- The select the Dispatch\_Date and Delivery\_Date from the Delivery table where Delivery\_ID is 511

-- The values will be stored in the 'v\_dispatch\_date' and 'v\_delivery\_date' variables.

SELECT Dispatch\_Date, Delivery\_Date INTO v\_dispatch\_date, v\_delivery\_date

FROM Delivery

WHERE Delivery\_ID = 511;

-- Checking to see if the dispatch date is after the delivery date, which is an invalid scenario

-- This is the condition where if true (invalid dates), the custom exception 'e\_invalid\_dates' is raised.

IF v\_dispatch\_date > v\_delivery\_date THEN

RAISE e\_invalid\_dates; -- Raise the user-defined exception for invalid dates

END IF;

-- The final output of the valid dispatch and delivery dates if no exception was raised

DBMS\_OUTPUT.PUT\_LINE('Dispatch Date: ' || v\_dispatch\_date);

DBMS\_OUTPUT.PUT\_LINE('Delivery Date: ' || v\_delivery\_date);

EXCEPTION

-- This will be handling the user-defined exception 'e\_invalid\_dates'

-- This block will be responsible in executing if the exception is raised, then it will be outputting the error message to the console.

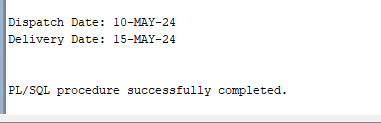
WHEN e\_invalid\_dates THEN

DBMS\_OUTPUT.PUT\_LINE('Error: Dispatch date cannot be after the delivery date.');

END;

/

**OUTPUT**

****

(IIE,2024)

**QUESTION 8**

(IIE,2024)

SELECT

c.First\_Name AS FIRST\_NAME, -- Displaying the correct customer's first name

c.Surname AS SURNAME, -- Display customer's surname

'R' || SUM(TO\_NUMBER(REPLACE(d.Price, 'R', ''))) AS AMOUNT, -- Total amount spent by a customer

CASE

WHEN SUM(TO\_NUMBER(REPLACE(d.Price, 'R', ''))) >= 1500 THEN '\*\*\*' -- 3 stars for total >= 1500

WHEN SUM(TO\_NUMBER(REPLACE(d.Price, 'R', ''))) BETWEEN 1000 AND 1499 THEN '\*\*' -- 2 stars for total between 1000 and 1499

ELSE '\*' -- 1 star for total < 1000

END AS CUSTOMER\_RATING -- This is where the Star rating is based on the total amount spent

FROM

Invoice i

JOIN

Customer c ON i.Customer\_ID = c.Customer\_ID

-- Joining the Customer and Invoice tables

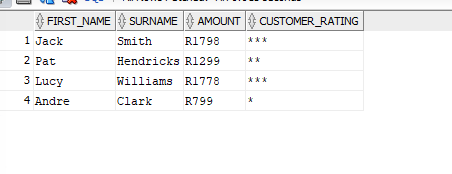
JOIN

Donation d ON i.Donation\_ID = d.Donation\_ID -- Joining the Donation and Invoice tables

GROUP BY

c.First\_Name, c.Surname -- Grouping the Customer's by first and last name

OUTPUT



(IIE,2024)

**Bibliography**

IIE, 2024. Advanced Databases Module Manual. 1st Edition ed. s.l.:The Indipendant Institute of Education.