Nikhil Saroop

ST10040092

ADD7311

ASSIGNMENT 2

Question 1

1. **Customer Table**

CREATE TABLE Customer (

Customer\_ID NUMBER PRIMARY KEY,

First\_Name VARCHAR2(50),

Surname VARCHAR2(50),

Address VARCHAR2(100),

Contact\_Number VARCHAR2(15),

Email VARCHAR2(50)

);

1. **Employee Table**

CREATE TABLE Employee (

Employee\_ID VARCHAR2(10) PRIMARY KEY,

First\_Name VARCHAR2(50),

Surname VARCHAR2(50),

Contact\_Number VARCHAR2(15),

Address VARCHAR2(100),

Email VARCHAR2(50)

);

1. **Donator Table**

CREATE TABLE Donator (

Donator\_ID NUMBER PRIMARY KEY,

First\_Name VARCHAR2(50),

Surname VARCHAR2(50),

Contact\_Number VARCHAR2(15),

Email VARCHAR2(50)

);

1. **Donation table**

CREATE TABLE Donation (

Donation\_ID NUMBER PRIMARY KEY,

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

Donation VARCHAR2(100),

Price NUMBER(10, 2), -- Store price as a NUMBER for proper calculations

Donation\_Date DATE

);

1. **Delivery Table**

CREATE TABLE Delivery (

Delivery\_ID NUMBER PRIMARY KEY,

Delivery\_Notes VARCHAR2(200),

Dispatch\_Date DATE,

Delivery\_Date DATE

);

1. **Returns Table**

CREATE TABLE Returns (

Return\_ID VARCHAR2(10) PRIMARY KEY,

Return\_Date DATE,

Reason VARCHAR2(200),

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

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

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

);

1. **Invoice Table**

CREATE TABLE Invoice (

Invoice\_Num NUMBER PRIMARY KEY,

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

Invoice\_Date DATE,

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

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

Delivery\_ID NUMBER REFERENCES Delivery(Delivery\_ID)

);

**Question 1 Populating Tables**

1. **Customer table**

**-- Insert data into the Customer table**

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');

1. **Employee Table**

**-- Insert data into the Employee table**

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');

1. **Donator Table**

**-- Insert data into the Donator table**

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');

**4. Donation Table**

**-- Insert data into the Donation table**

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'));

1. **Delievery Table**

**-- Insert data into the Delivery table**

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'));

**6. Returns Table**

**-- Insert data into the Returns table**

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');

**7. Invoice Table**

**-- Insert data into the Invoice table**

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);

**Showing Tables**

SELECT \* FROM Employee;

SELECT \* FROM Donator;

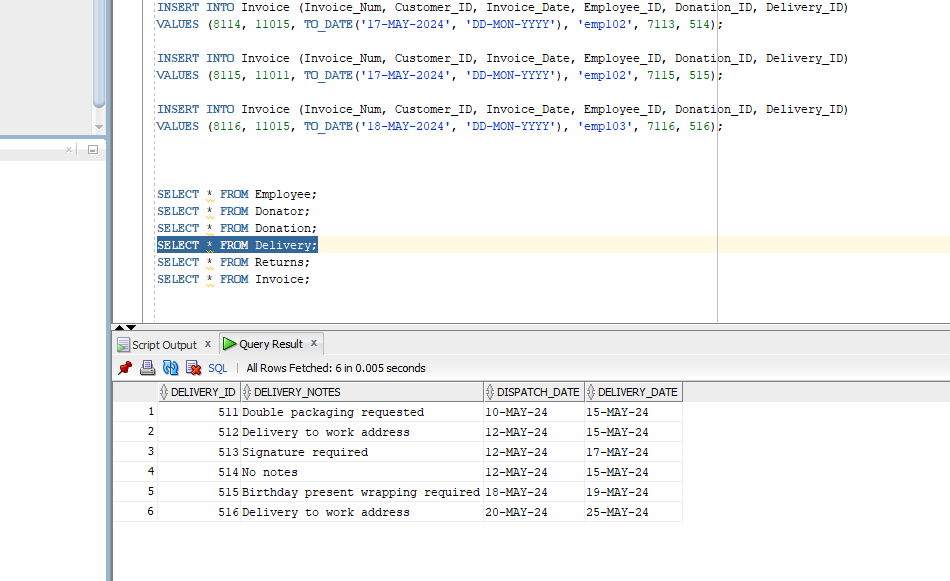
SELECT \* FROM Donation;

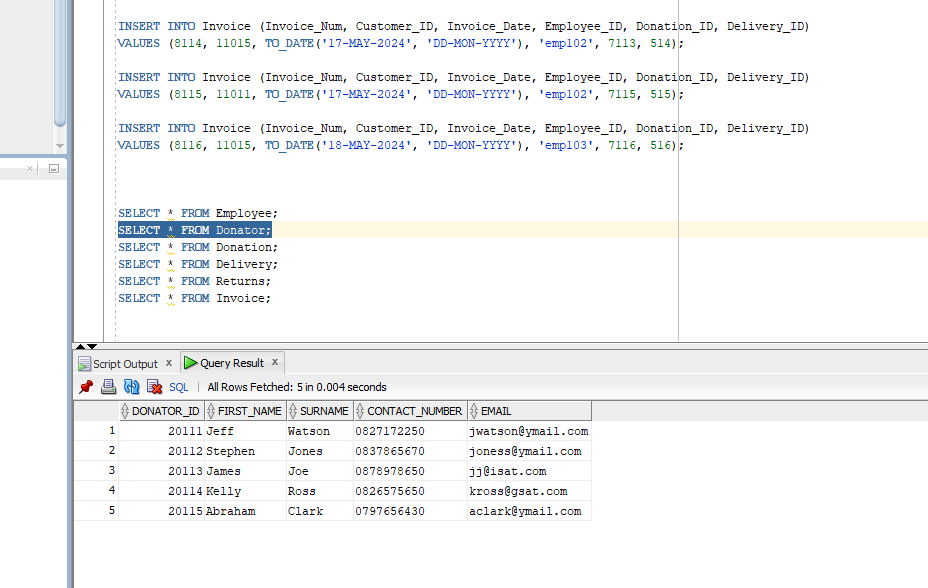
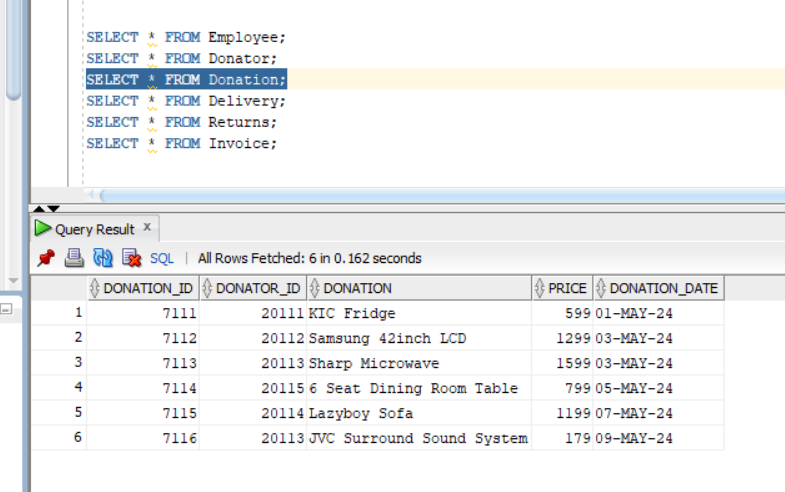
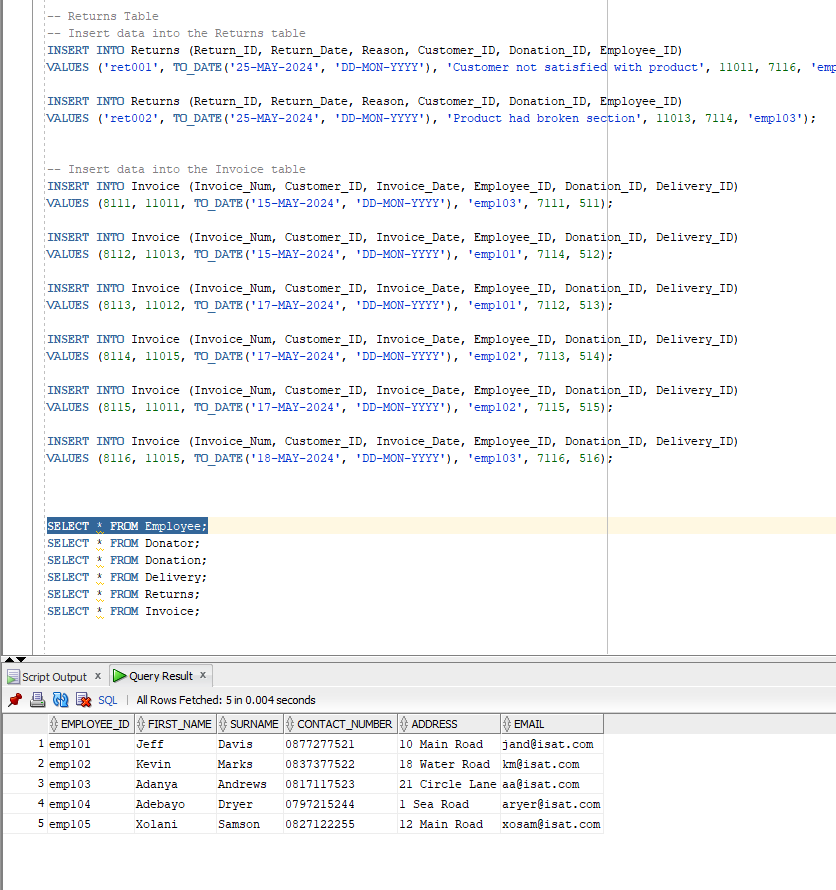
SELECT \* FROM Delivery;

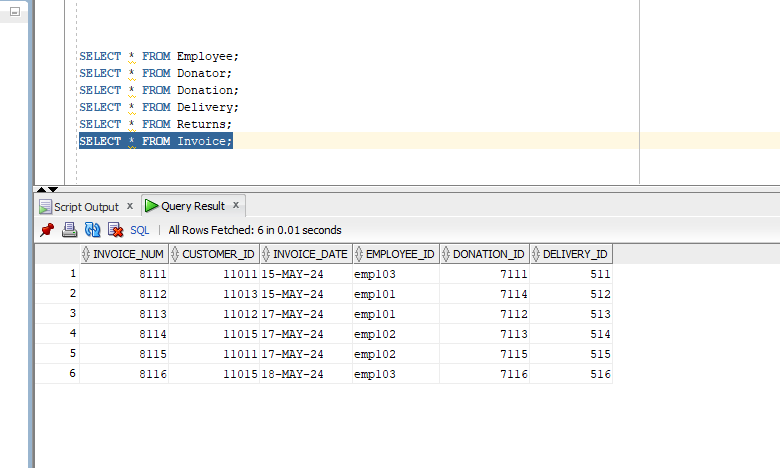
SELECT \* FROM Returns;

SELECT \* FROM Invoice;

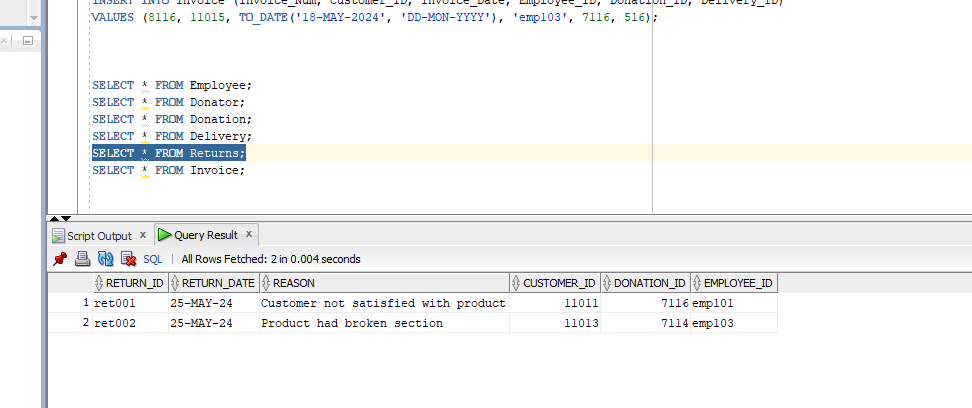
1. **Delivery Table**



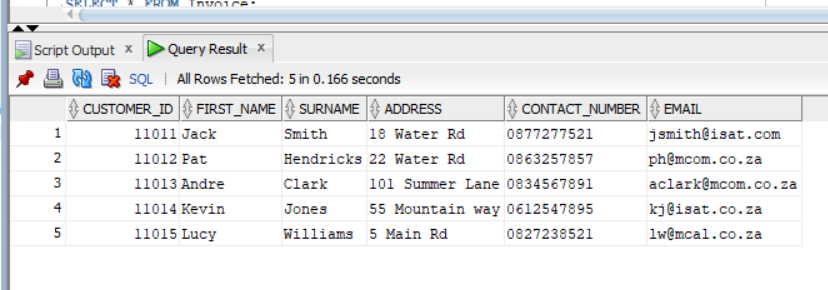
1. **Donator Table**
2. **Donation**
3. **Employee Table**
4. **Invoice table**



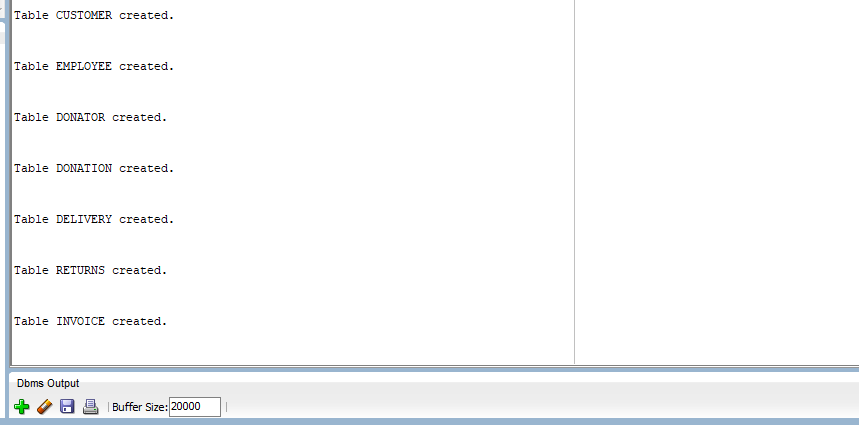
1. **Returns Table**



**7. Customer Table**



Console OUTPUT



**Question 2**

SELECT

c.First\_Name || ', ' || c.Surname AS Customer\_Name, -- Concatenating first name and surname with a comma

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

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

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

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

i.Invoice\_Date -- Invoice date to show when filtering

FROM

Invoice i

JOIN

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

JOIN

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

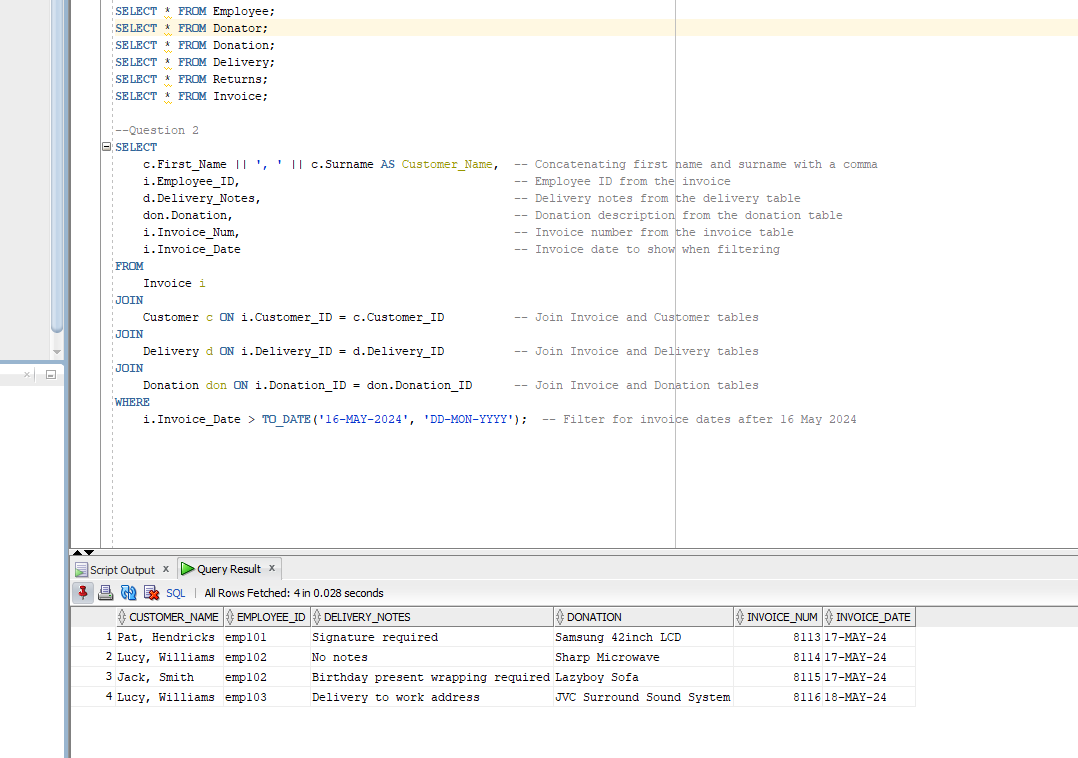
JOIN

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

WHERE

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

**OUTPUT**



**Question 3**

**Step 1:**

**-- Create the Funding table**

CREATE TABLE Funding (

funding\_id NUMBER GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY, -- Auto-generated unique ID

funder VARCHAR2(100), -- Name of the funder

funding\_amount NUMBER(10, 2) -- Amount of funding

);

**--The funding\_id column uses GENERATED BY DEFAULT AS IDENTITY to automatically generate a unique ID for each inserted record.**

**--The funder column is a VARCHAR2(100) to store the organization's name or person providing the funding.**

**--The funding\_amount column is a NUMBER(10, 2) to store the funding amount, allowing up to 10 digits with 2 decimal places for precision (e.g., 50000.00).**

**Step 2: Example Insert Statements**

**-- Insert a new funding record**

INSERT INTO Funding (funder, funding\_amount)

VALUES ('United Charity', 50000.00);

**-- Insert another funding record**

INSERT INTO Funding (funder, funding\_amount)

VALUES ('Global Aid', 75000.50);

**-- Insert a third record**

INSERT INTO Funding (funder, funding\_amount)

VALUES ('Helping Hands Foundation', 25000.00);

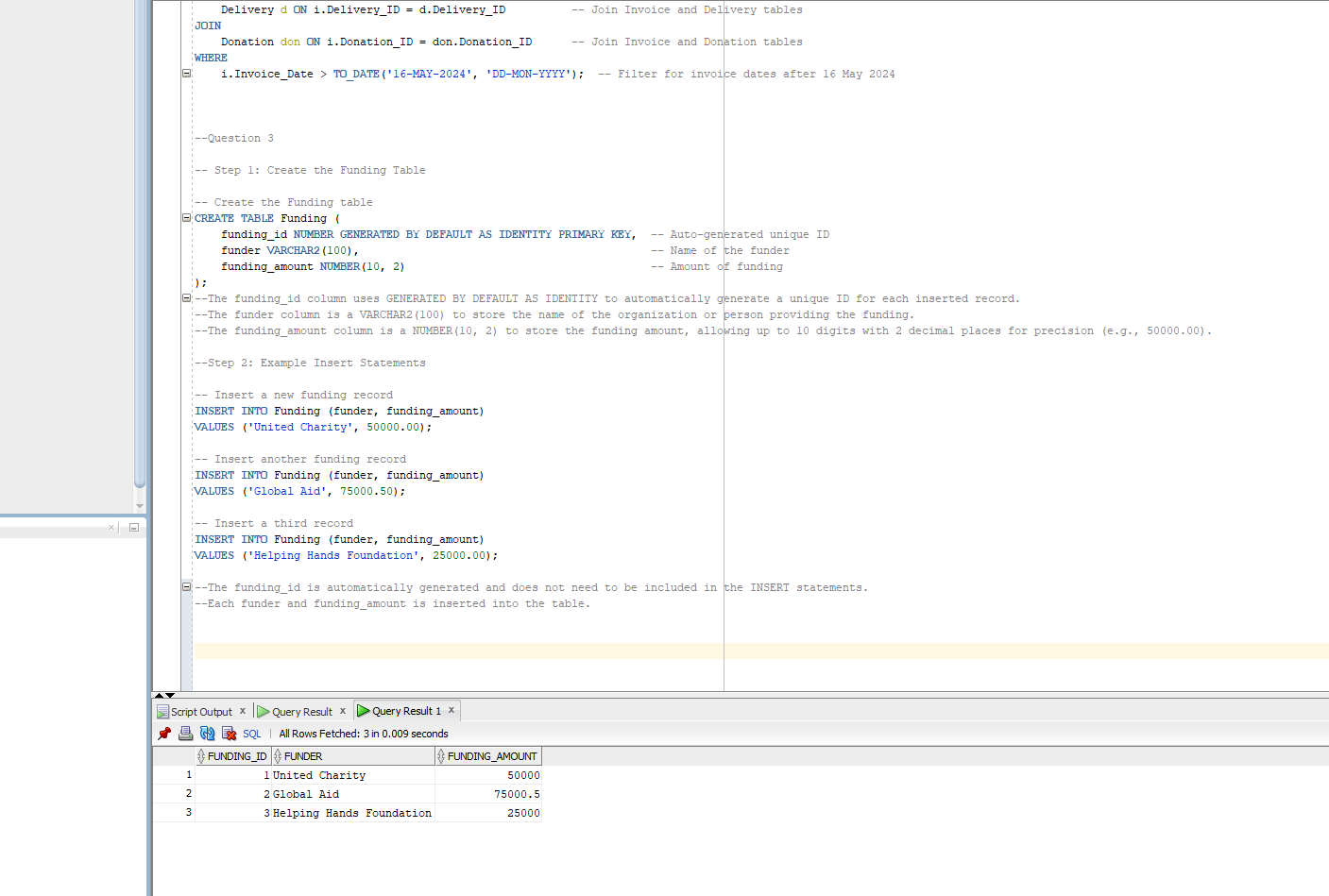
--The funding\_id is automatically generated and does not need to be included in the INSERT statements.

--Each funder and funding\_amount is inserted into the table.

**--Step 3: Query to View the Data**

**-- Query to view all the funding records**

SELECT \* FROM Funding;



**Question 4**

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, -- Donation description

d.Price, -- Donation price

ret.Reason -- Reason for return

FROM

Returns ret

JOIN

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

JOIN

Donation d ON ret.Donation\_ID = d.Donation\_ID -- 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;

/

**--Explanation:**

**--SET SERVEROUTPUT ON: This command ensures that the output from the PL/SQL block is displayed in the console.**

**--Cursor (FOR Loop): The cursor retrieves data from the Returns, Customer, and Donation tables, joining them based on the foreign keys to display the necessary information.**

**--c.First\_Name || ', ' || c.Surname AS Customer\_Name: This concatenates the customer’s first name and surname with a comma.**

**--d.Donation: The donation purchased by the customer.**

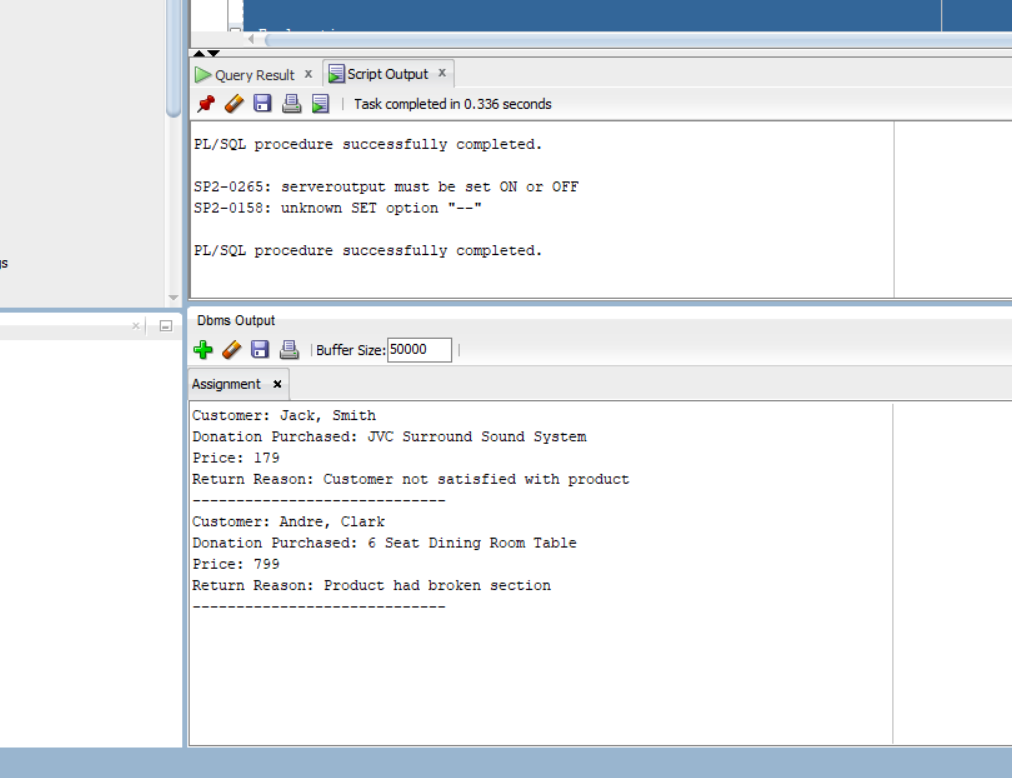
**--d.Price: The price of the donation.**

**--ret.Reason: The reason for the return from the Returns table.**

**--DBMS\_OUTPUT.PUT\_LINE: This is used to print each piece of information (customer name, donation, price, reason) in a formatted manner.**

**--END LOOP: The loop iterates over each result from the cursor and prints the details.**

**OUTPUT**



Question 5

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, -- Customer First Name

c.Surname, -- 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 -- Join with Customer table

JOIN

Employee e ON i.Employee\_ID = e.Employee\_ID -- Join with Employee table

JOIN

Donation d ON i.Donation\_ID = d.Donation\_ID -- Join with Donation table

JOIN

Delivery del ON i.Delivery\_ID = del.Delivery\_ID -- Join with Delivery table

WHERE

i.Customer\_ID = 11011 -- 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);

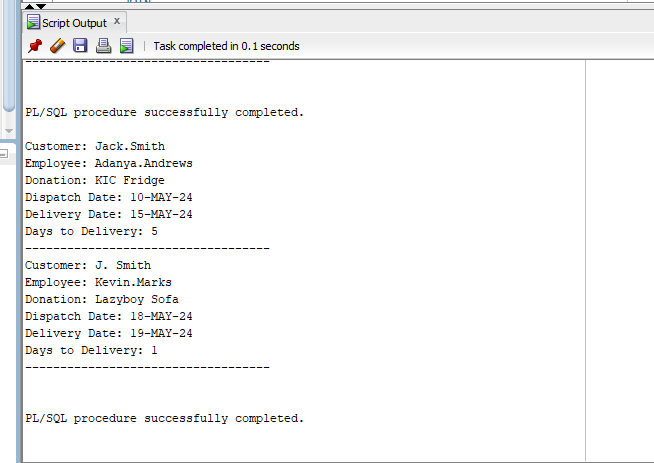
DBMS\_OUTPUT.PUT\_LINE('-----------------------------------');

END LOOP;

END;

END;

/



(IIE,2024)

**Question 6**

SET SERVEROUTPUT ON;

BEGIN

-- Cursor to fetch customer names and their total amount spent on donations

FOR rec IN (

SELECT

c.First\_Name || ', ' || c.Surname AS Customer\_Name, -- Concatenate First Name and Surname

SUM(TO\_NUMBER(REPLACE(d.Price, 'R', ''))) AS Total\_Spent -- Remove "R" and convert Price to number

FROM

Invoice i

JOIN

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

JOIN

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

GROUP BY

c.First\_Name, c.Surname -- Group by Customer to sum the total spent

)

LOOP

-- Output the customer name

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

-- Determine the output based on total spent

IF rec.Total\_Spent >= 1500 THEN

-- Display the amount and 3-star rating for totals >= 1500

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

ELSE

-- Display the amount only for totals < 1500 (no stars, no brackets)

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

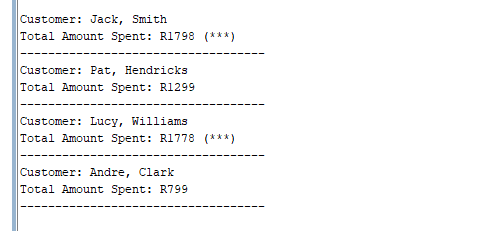
END IF;

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

END LOOP;

END;

/



(IIE,2024)

Question 7

Q7.1

SET SERVEROUTPUT ON;

DECLARE

v\_customer\_email Customer.Email%TYPE; -- Declare variable with the same datatype as Email in Customer table

BEGIN

-- Select the email of the customer with ID 11012

SELECT Email INTO v\_customer\_email

FROM Customer

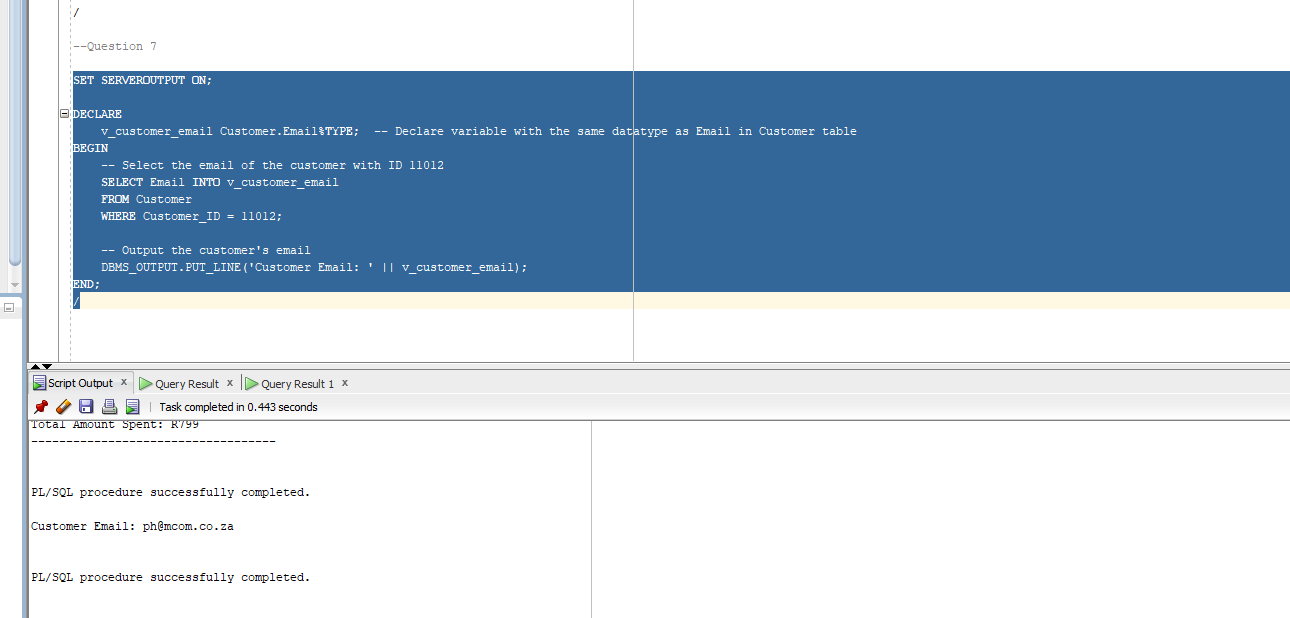
WHERE Customer\_ID = 11012;

-- Output the customer's email

DBMS\_OUTPUT.PUT\_LINE('Customer Email: ' || v\_customer\_email);

END;

/



Q7.2

SET SERVEROUTPUT ON;

DECLARE

v\_invoice Invoice%ROWTYPE; -- Declare a variable with the same structure as a row in the Invoice table

BEGIN

-- Select the entire row for the invoice of customer with ID 11012

SELECT \* INTO v\_invoice

FROM Invoice

WHERE Customer\_ID = 11012;

-- Output the invoice details

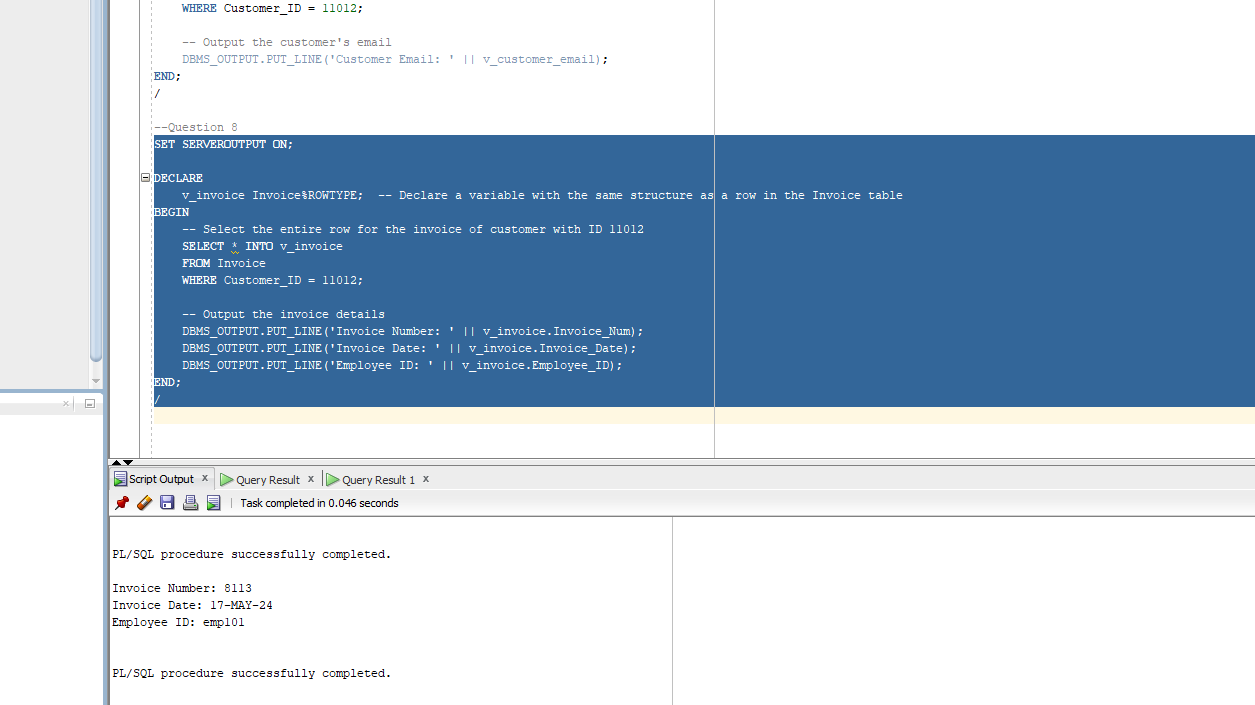
DBMS\_OUTPUT.PUT\_LINE('Invoice Number: ' || v\_invoice.Invoice\_Num);

DBMS\_OUTPUT.PUT\_LINE('Invoice Date: ' || v\_invoice.Invoice\_Date);

DBMS\_OUTPUT.PUT\_LINE('Employee ID: ' || v\_invoice.Employee\_ID);

END;

/



(IIE,2024)

Q7.3

SET SERVEROUTPUT ON;

DECLARE

-- Declare a user-defined exception for low total spending

e\_low\_spending EXCEPTION;

v\_total\_spent NUMBER; -- Declare a variable to hold the total amount spent by a customer

BEGIN

-- Select the total amount spent by customer with ID 11012

SELECT SUM(TO\_NUMBER(REPLACE(Price, 'R', ''))) INTO v\_total\_spent

FROM Invoice i

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

WHERE i.Customer\_ID = 11012;

-- Check if the total amount spent is less than R500

IF v\_total\_spent < 500 THEN

RAISE e\_low\_spending; -- Raise the custom exception

END IF;

-- Output the total spent

DBMS\_OUTPUT.PUT\_LINE('Total Amount Spent: R' || v\_total\_spent);

EXCEPTION

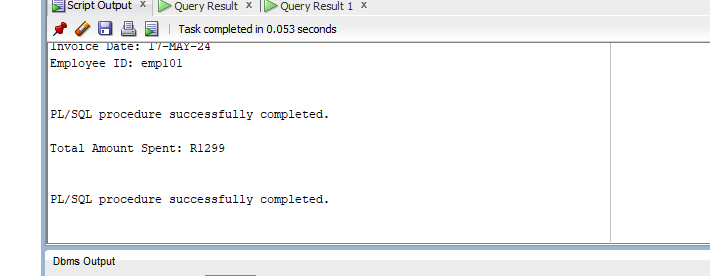
-- Handle the user-defined exception

WHEN e\_low\_spending THEN

DBMS\_OUTPUT.PUT\_LINE('Error: Total amount spent is less than R500.');

END;

/



(IIE,2024)

Question 8

SELECT

c.First\_Name AS FIRST\_NAME, -- Display 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 the 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 -- Star rating based on the total amount spent

FROM

Invoice i

JOIN

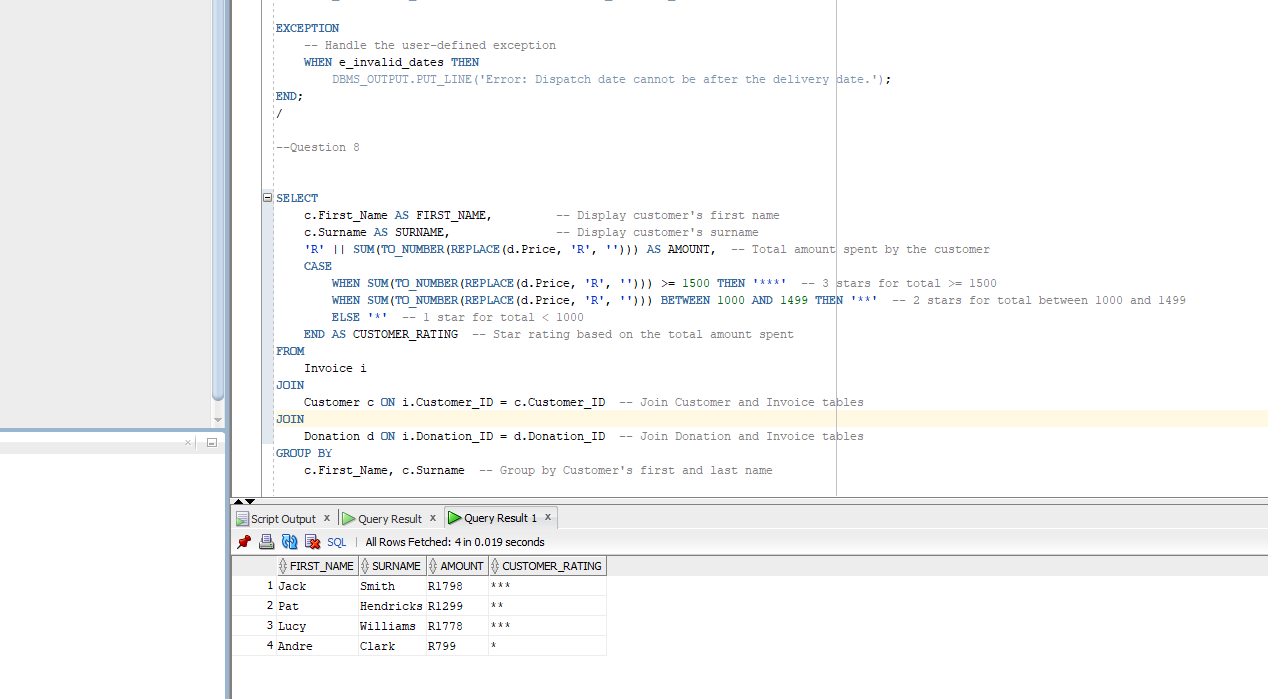
Customer c ON i.Customer\_ID = c.Customer\_ID -- Join Customer and Invoice tables

JOIN

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

GROUP BY

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



(IIE,2024)

Bibliography

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