7/10/2024

ADDB 7311

ASSIGNMENT 2

Sajana Bidesi

ST10249843

# CONTENTS

[CONTENTS 1](#_Toc179204664)

[QUESTION 1 2](#_Toc179204665)

[CREATING THE TABLES 2](#_Toc179204666)

[POPULATING THE TABLES 5](#_Toc179204667)

[QUESTION 2 10](#_Toc179204668)

[QUESTION 3 11](#_Toc179204669)

[QUESTION 4 13](#_Toc179204670)

[QUESTION 5 15](#_Toc179204671)

[QUESTION 6 17](#_Toc179204672)

[QUESTION 7 19](#_Toc179204673)

[QUESTION 7.1 19](#_Toc179204674)

[QUESTION 7.2 21](#_Toc179204675)

[QUESTION 7.3 23](#_Toc179204676)

[QUESTION 8 25](#_Toc179204677)

[REFERENCES 26](#_Toc179204678)

# QUESTION 1

## CREATING THE TABLES

-- CREATE TABLE CUSTOMER

CREATE TABLE CUSTOMER

(

CUSTOMER\_ID NUMBER PRIMARY KEY NOT NULL,

FIRST\_NAME VARCHAR(50) NOT NULL,

SURNAME VARCHAR(50) NOT NULL,

ADDRESS VARCHAR(100) NOT NULL,

CONTACT\_NUMBER VARCHAR(50) NOT NULL,

EMAIL VARCHAR(50) NOT NULL

);

-- CREATE TABLE EMPLOYEE

CREATE TABLE EMPLOYEE

(

EMPLOYEE\_ID VARCHAR(50) PRIMARY KEY NOT NULL,

FISRT\_NAME VARCHAR(50) NOT NULL,

SURNAME VARCHAR(50) NOT NULL,

CONTACT\_NUMBER VARCHAR(50) NOT NULL,

ADDRESS VARCHAR(100) NOT NULL,

EMAIL VARCHAR(50) NOT NULL

);

-- CREATE TABLE DONATOR

CREATE TABLE DONATOR

(

DONATOR\_ID NUMBER PRIMARY KEY NOT NULL,

FIRST\_NAME VARCHAR(50) NOT NULL,

SURNAME VARCHAR(50) NOT NULL,

CONTACT\_NUMBER VARCHAR(50) NOT NULL,

EMAIL VARCHAR(50) NOT NULL

);

-- CREATE TABLE DONATION

CREATE TABLE DONATION

(

DONATION\_ID NUMBER PRIMARY KEY NOT NULL,

DONATOR\_ID NUMBER NOT NULL,

FOREIGN KEY (DONATOR\_ID) REFERENCES DONATOR(DONATOR\_ID),

DONATION VARCHAR(100) NOT NULL,

PRICE NUMBER NOT NULL,

DONATION\_DATE DATE NOT NULL

);

-- CREATE TABLE DELIVERY

CREATE TABLE DELIVERY

(

DELIVERY\_ID NUMBER PRIMARY KEY NOT NULL,

DELIVERY\_NOTES VARCHAR(100) NOT NULL,

DISPATCH\_DATE DATE NOT NULL,

DELIVERY\_DATE DATE NOT NULL

);

-- CREATE TABLE RETURNS

CREATE TABLE RETURNS

(

RETURN\_ID NUMBER PRIMARY KEY NOT NULL,

RETURN\_DATE DATE NOT NULL,

REASON VARCHAR(100) NOT NULL,

CUSTOMER\_ID NUMBER NOT NULL,

FOREIGN KEY (CUSTOMER\_ID) REFERENCES CUSTOMER(CUSTOMER\_ID),

DONATION\_ID NUMBER NOT NULL,

FOREIGN KEY (DONATION\_ID) REFERENCES DONATION(DONATION\_ID),

EMPLOYEE\_ID VARCHAR(50) NOT NULL,

FOREIGN KEY (EMPLOYEE\_ID) REFERENCES EMPLOYEE(EMPLOYEE\_ID)

);

-- CREATE TABLE INVOICE

CREATE TABLE INVOICE

(

INVOICE\_NUM NUMBER PRIMARY KEY NOT NULL,

CUSTOMER\_ID NUMBER NOT NULL,

FOREIGN KEY (CUSTOMER\_ID) REFERENCES CUSTOMER(CUSTOMER\_ID),

INVOICE\_DATE DATE NOT NULL,

EMPLOYEE\_ID VARCHAR(50) NOT NULL,

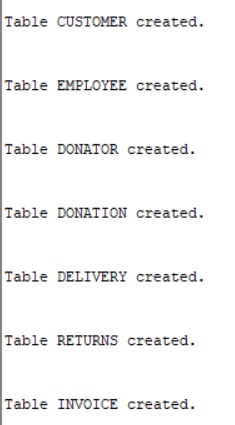
FOREIGN KEY (EMPLOYEE\_ID) REFERENCES EMPLOYEE(EMPLOYEE\_ID),

DONATION\_ID NUMBER NOT NULL,

FOREIGN KEY (DONATION\_ID) REFERENCES DONATION(DONATION\_ID),

DELIVERY\_ID NUMBER NOT NULL,

FOREIGN KEY (DELIVERY\_ID) REFERENCES DELIVERY(DELIVERY\_ID)

);

## POPULATING THE TABLES

-- INSERT INTO CUSTOMER

INSERT ALL

INTO CUSTOMER (CUSTOMER\_ID, FIRST\_NAME, SURNAME, ADDRESS, CONTACT\_NUMBER, EMAIL) VALUES (11011, 'Jack', 'Smith', '18 Water Rd', '0877277521', 'jsmith@isat.com')

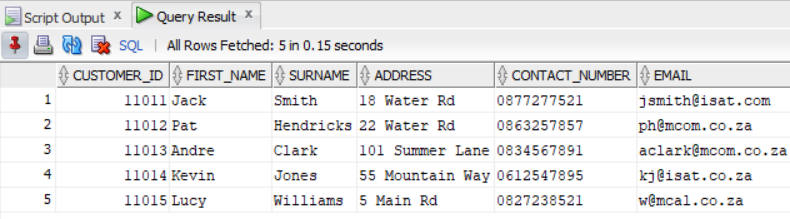
INTO CUSTOMER (CUSTOMER\_ID, FIRST\_NAME, SURNAME, ADDRESS, CONTACT\_NUMBER, EMAIL) VALUES (11012, 'Pat', 'Hendricks', '22 Water Rd', '0863257857', 'ph@mcom.co.za')

INTO CUSTOMER (CUSTOMER\_ID, FIRST\_NAME, SURNAME, ADDRESS, CONTACT\_NUMBER, EMAIL) VALUES (11013, 'Andre', 'Clark', '101 Summer Lane', '0834567891', 'aclark@mcom.co.za')

INTO CUSTOMER (CUSTOMER\_ID, FIRST\_NAME, SURNAME, ADDRESS, CONTACT\_NUMBER, EMAIL) VALUES (11014, 'Kevin', 'Jones', '55 Mountain Way', '0612547895', 'kj@isat.co.za')

INTO CUSTOMER (CUSTOMER\_ID, FIRST\_NAME, SURNAME, ADDRESS, CONTACT\_NUMBER, EMAIL) VALUES (11015, 'Lucy', 'Williams', '5 Main Rd', '0827238521', 'w@mcal.co.za')

SELECT \* FROM dual;

SELECT \* FROM CUSTOMER;

-- INSERT INTO EMPLOYEE

INSERT ALL

INTO EMPLOYEE (EMPLOYEE\_ID, FISRT\_NAME, SURNAME, CONTACT\_NUMBER, ADDRESS, EMAIL) VALUES ('emp101', 'Jeff', 'Davis', '0877277521', '10 Main Rd', 'jand@isat.com')

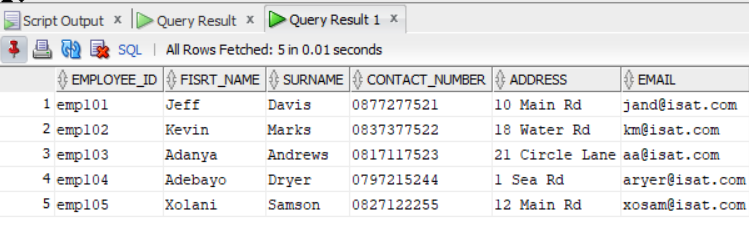
INTO EMPLOYEE (EMPLOYEE\_ID, FISRT\_NAME, SURNAME, CONTACT\_NUMBER, ADDRESS, EMAIL) VALUES ('emp102', 'Kevin', 'Marks', '0837377522', '18 Water Rd', 'km@isat.com')

INTO EMPLOYEE (EMPLOYEE\_ID, FISRT\_NAME, SURNAME, CONTACT\_NUMBER, ADDRESS, EMAIL) VALUES ('emp103', 'Adanya', 'Andrews', '0817117523', '21 Circle Lane', 'aa@isat.com')

INTO EMPLOYEE (EMPLOYEE\_ID, FISRT\_NAME, SURNAME, CONTACT\_NUMBER, ADDRESS, EMAIL) VALUES ('emp104', 'Adebayo', 'Dryer', '0797215244', '1 Sea Rd', 'aryer@isat.com')

INTO EMPLOYEE (EMPLOYEE\_ID, FISRT\_NAME, SURNAME, CONTACT\_NUMBER, ADDRESS, EMAIL) VALUES ('emp105', 'Xolani', 'Samson', '0827122255', '12 Main Rd', 'xosam@isat.com')

SELECT \* FROM dual;

SELECT \* FROM EMPLOYEE;

-- INSERT INTO DONATOR

INSERT ALL

INTO DONATOR (DONATOR\_ID, FIRST\_NAME, SURNAME, CONTACT\_NUMBER, EMAIL) VALUES (20111, 'Jeff', 'Watson', '0827172250', 'jwatson@ymail.com')

INTO DONATOR (DONATOR\_ID, FIRST\_NAME, SURNAME, CONTACT\_NUMBER, EMAIL) VALUES (20112, 'Stephen', 'Jones', '0837865670', 'joness@ymail.com')

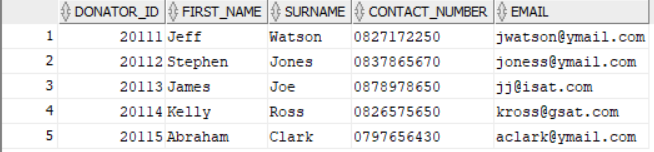
INTO DONATOR (DONATOR\_ID, FIRST\_NAME, SURNAME, CONTACT\_NUMBER, EMAIL) VALUES (20113, 'James', 'Joe', '0878978650', 'jj@isat.com')

INTO DONATOR (DONATOR\_ID, FIRST\_NAME, SURNAME, CONTACT\_NUMBER, EMAIL) VALUES (20114, 'Kelly', 'Ross', '0826575650', 'kross@gsat.com')

INTO DONATOR (DONATOR\_ID, FIRST\_NAME, SURNAME, CONTACT\_NUMBER, EMAIL) VALUES (20115, 'Abraham', 'Clark', '0797656430', 'aclark@ymail.com')

SELECT \* FROM dual;

SELECT \* FROM DONATOR;



-- INSERT INTO DONATIONS

INSERT ALL

INTO DONATION (DONATION\_ID, DONATOR\_ID, DONATION, PRICE, DONATION\_DATE) VALUES (7111, 20111, 'KIC Fridge', 599, TO\_DATE('01-May-2024', 'DD-Mon-YYYY'))

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'))

INTO DONATION (DONATION\_ID, DONATOR\_ID, DONATION, PRICE, DONATION\_DATE) VALUES (7113, 20113, 'Sharp Microwave', 1599, TO\_DATE('03-May-2024', 'DD-Mon-YYYY'))

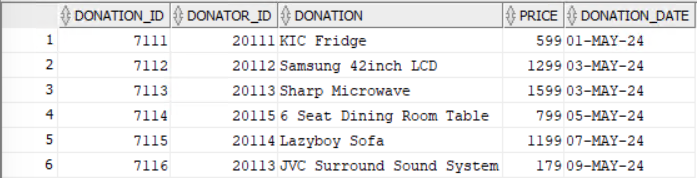
INTO DONATION (DONATION\_ID, DONATOR\_ID, DONATION, PRICE, DONATION\_DATE) VALUES (7114, 20115, '6 Seat Dining Room Table', 799, TO\_DATE('05-May-2024', 'DD-Mon-YYYY'))

INTO DONATION (DONATION\_ID, DONATOR\_ID, DONATION, PRICE, DONATION\_DATE) VALUES (7115, 20114, 'Lazyboy Sofa', 1199, TO\_DATE('07-May-2024', 'DD-Mon-YYYY'))

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'))

SELECT \* FROM dual;

SELECT \* FROM DONATION;



-- INSERT INTO DELIVERY

INSERT ALL

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'))

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'))

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'))

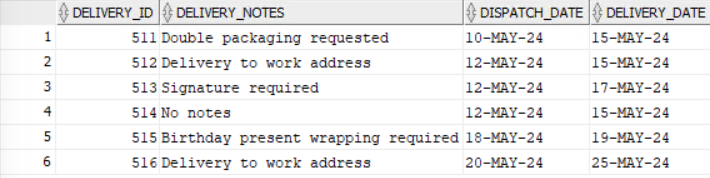
INTO DELIVERY (DELIVERY\_ID, DELIVERY\_NOTES, DISPATCH\_DATE, DELIVERY\_DATE) VALUES (514, 'No notes', TO\_DATE('12-May-2024', 'DD-Mon-YYYY'), TO\_DATE('15-May-2024', 'DD-Mon-YYYY'))

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'))

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'))

SELECT \* FROM dual;

SELECT \* FROM DELIVERY;



-- INSERT INTO RETURNS

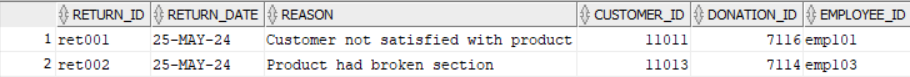
INSERT ALL

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')

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')

SELECT \* FROM dual;

SELECT \* FROM RETURNS;



INSERT ALL

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)

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)

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)

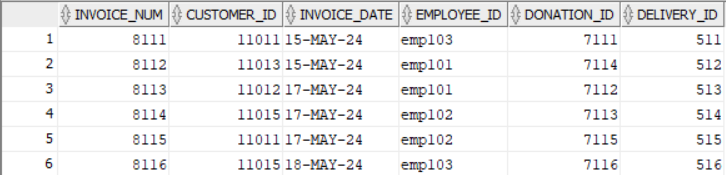
INTO INVOICE (INVOICE\_NUM, CUSTOMER\_ID, INVOICE\_DATE, EMPLOYEE\_ID, DONATION\_ID, DELIVERY\_ID) VALUES (8114, 11015, TO\_DATE('17-May-2024', 'DD-Mon-YYYY'), 'emp102', 7113, 514)

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)

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)

SELECT \* FROM dual;

SELECT \* FROM INVOICE;



# QUESTION 2

-- QUESTION 2

SELECT

CUST.FIRST\_NAME || ',' || CUST.SURNAME AS CUSTOMER\_NAME,

EMP.EMPLOYEE\_ID,

DEL.DELIVERY\_NOTES,

DON.DONATION,

INV.INVOICE\_NUM,

INV.INVOICE\_DATE

FROM

CUSTOMER CUST

JOIN INVOICE INV ON CUST.CUSTOMER\_ID = INV.CUSTOMER\_ID

JOIN EMPLOYEE EMP ON INV.EMPLOYEE\_ID = EMP.EMPLOYEE\_ID

JOIN DONATION DON ON INV.DONATION\_ID = DON.DONATION\_ID

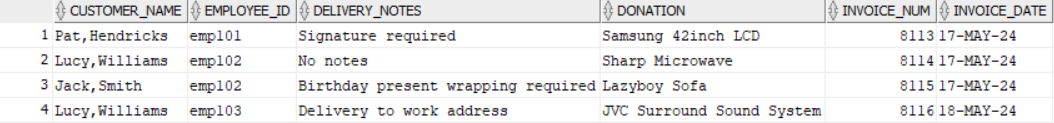
JOIN DELIVERY DEL ON INV.DELIVERY\_ID = DEL.DELIVERY\_ID

WHERE

INV.INVOICE\_DATE > TO\_DATE('16-MAY-2024', 'DD-MON-YYYY')

ORDER BY

DON.DONATION\_DATE;



# QUESTION 3

-- QUESTION 3 (The IIE, 2024)

-- CREATE THE SEQUENCE TO INCREMENT THE VALUE FOR THE ID

CREATE SEQUENCE funding\_seq

START WITH 1

INCREMENT BY 1

NOCACHE;

-- CREATE THE FUNDING TABLE WITH THE PROVIDED ATTRIBUTES

CREATE TABLE FUNDING (

FUNDING\_ID INT PRIMARY KEY,

FUNDER VARCHAR(255) NOT NULL,

FUNDING\_AMOUNT NUMBER NOT NULL

);

-- INSERT A RECORT USING THE SEQUENCE

INSERT INTO Funding (FUNDING\_ID, FUNDER, FUNDING\_AMOUNT)

VALUES (funding\_seq.NEXTVAL, 'Sajana Bidesi', 1000);

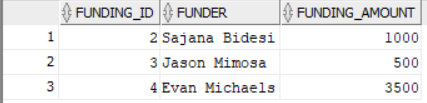
INSERT INTO Funding (FUNDING\_ID, FUNDER, FUNDING\_AMOUNT)

VALUES (funding\_seq.NEXTVAL, 'Jason Mimosa', 500);

INSERT INTO Funding (FUNDING\_ID, FUNDER, FUNDING\_AMOUNT)

VALUES (funding\_seq.NEXTVAL, 'Evan Michaels', 3500);

SELECT \* FROM FUNDING;



In Oracle databases, a sequence is a database object that generates a sequential series of unique numbers, which can be used primarily for creating primary keys or unique identifiers for rows in a table. When you create a sequence, you define its starting point, the increment value, and other optional parameters like caching and maximum values. Sequences are not tied to any specific table, allowing them to generate unique values across different tables. Each time a sequence is referenced (typically using `NEXTVAL`), it produces the next number in the sequence, ensuring that no two rows receive the same identifier, which helps maintain data integrity and uniqueness within the database. (Oracle, 2019; The IIE, 2024)

# QUESTION 4

-- QUESTION 4

SET SERVEROUTPUT ON;

DECLARE

CURSOR c\_donations IS

SELECT

c.FIRST\_NAME || ' ' || c.SURNAME AS Customer,

d.DONATION,

d.PRICE,

r.REASON AS "Return Reason"

FROM

CUSTOMER c

JOIN

RETURNS r ON c.CUSTOMER\_ID = r.CUSTOMER\_ID

JOIN

DONATION d ON r.DONATION\_ID = d.DONATION\_ID;

BEGIN

FOR rec IN c\_donations LOOP

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

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

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

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

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

END LOOP;

END;

/

A screenshot of a computer

Description automatically generated

# QUESTION 5

-- QUESTION 5(The IIE, 2024)

SET SERVEROUTPUT ON;

BEGIN

FOR rec IN (

SELECT

SUBSTR(c.FIRST\_NAME, 1, 1) || '.' AS customer\_initial,

c.SURNAME AS customer\_surname,

SUBSTR(e.FISRT\_NAME, 1, 1) || '.' || e.SURNAME AS employee\_name,

d.DONATION AS donation,

del.DISPATCH\_DATE AS dispatch\_date,

del.DELIVERY\_DATE AS delivery\_date,

(del.DELIVERY\_DATE - del.DISPATCH\_DATE) AS days\_to\_deliver

FROM

CUSTOMER c

JOIN

INVOICE i ON c.CUSTOMER\_ID = i.CUSTOMER\_ID

JOIN

EMPLOYEE e ON i.EMPLOYEE\_ID = e.EMPLOYEE\_ID

JOIN

DONATION d ON i.DONATION\_ID = d.DONATION\_ID

JOIN

DELIVERY del ON i.DELIVERY\_ID = del.DELIVERY\_ID

WHERE

c.CUSTOMER\_ID = 11011

) LOOP

DBMS\_OUTPUT.PUT\_LINE('Customer: ' || rec.customer\_initial || ' ' || rec.customer\_surname);

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);

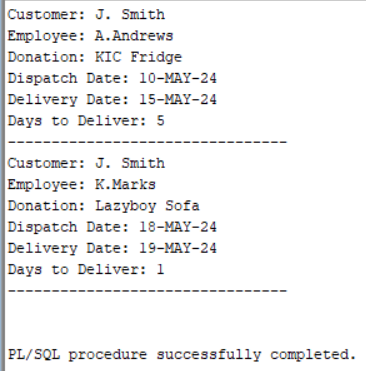
DBMS\_OUTPUT.PUT\_LINE('Days to Deliver: ' || rec.days\_to\_deliver);

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

END LOOP;

END;

/



# QUESTION 6

-- QUESTION 6(The IIE, 2024)

SET SERVEROUTPUT ON;

BEGIN

FOR rec IN (

SELECT

c.FIRST\_NAME,

c.SURNAME,

SUM(d.PRICE) AS TotalSpent,

CASE

WHEN SUM(d.PRICE) >= 1500 THEN '(\*\*\*)'

ELSE ''

END AS Rating

FROM CUSTOMER c

JOIN INVOICE i ON c.CUSTOMER\_ID = i.CUSTOMER\_ID

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

WHERE c.FIRST\_NAME IN ('Jack', 'Lucy', 'Pat', 'Andre')

GROUP BY c.FIRST\_NAME, c.SURNAME

)

LOOP

DBMS\_OUTPUT.PUT\_LINE('First Name: ' || rec.FIRST\_NAME);

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

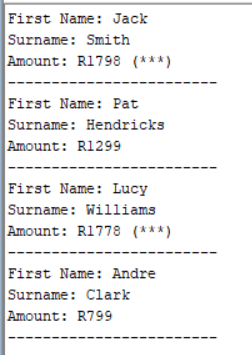
DBMS\_OUTPUT.PUT\_LINE('Amount: R' || rec.TotalSpent || ' ' || rec.Rating);

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

END LOOP;

END;

/



# QUESTION 7

## QUESTION 7.1

-- QUESTION 7.1(Moore et al., 2019)

DECLARE

-- DECLARE VARIABLES USING %TYPE ATTRIBUTE

v\_customer\_id CUSTOMER.CUSTOMER\_ID%TYPE;

v\_first\_name CUSTOMER.FIRST\_NAME%TYPE;

v\_surname CUSTOMER.SURNAME%TYPE;

v\_address CUSTOMER.ADDRESS%TYPE;

v\_contact\_number CUSTOMER.CONTACT\_NUMBER%TYPE;

v\_email CUSTOMER.EMAIL%TYPE;

BEGIN

-- ASSIGN A SAMPLE CUSTOMER ID FOR FETCHING DETAILS

v\_customer\_id := 11011;

-- FETCH CUSTOMER DETAILS INTO VARIABLES

SELECT FIRST\_NAME, SURNAME, ADDRESS, CONTACT\_NUMBER, EMAIL

INTO v\_first\_name, v\_surname, v\_address, v\_contact\_number, v\_email

FROM CUSTOMER

WHERE CUSTOMER\_ID = v\_customer\_id;

-- OUTPUT CUSTOMER DETAILS

DBMS\_OUTPUT.PUT\_LINE('Customer ID: ' || v\_customer\_id);

DBMS\_OUTPUT.PUT\_LINE('First Name: ' || v\_first\_name);

DBMS\_OUTPUT.PUT\_LINE('Surname: ' || v\_surname);

DBMS\_OUTPUT.PUT\_LINE('Address: ' || v\_address);

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

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

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

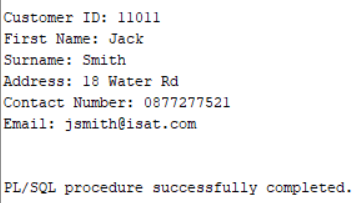
DBMS\_OUTPUT.PUT\_LINE('No customer found with ID: ' || v\_customer\_id);

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('An error occurred: ' || SQLERRM);

END;

/



## QUESTION 7.2

-- QUESTION 7.2(L. Jayapalan et al., 2020)

DECLARE

-- DECLARE A RECORD VARIABLE THAT CAN HOLD A ROW FROM THE CUSTOMER TABLE

customer\_record CUSTOMER%ROWTYPE;

-- DECLARE A CURSOR TO SELECT ALL CUSTOMERS

CURSOR customer\_cursor IS

SELECT \* FROM CUSTOMER;

BEGIN

OPEN customer\_cursor;

-- LOOP THROUGH EACH ROW RETURNED BY THE CURSOR

LOOP

FETCH customer\_cursor INTO customer\_record;

-- EXIT THE LOOP WHEN NO MORE ROWS ARE FOUND

EXIT WHEN customer\_cursor%NOTFOUND;

-- DISPLAY THE CUSTOMER DETAILS

DBMS\_OUTPUT.PUT\_LINE('Customer ID: ' || customer\_record.CUSTOMER\_ID);

DBMS\_OUTPUT.PUT\_LINE('First Name: ' || customer\_record.FIRST\_NAME);

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

DBMS\_OUTPUT.PUT\_LINE('Address: ' || customer\_record.ADDRESS);

DBMS\_OUTPUT.PUT\_LINE('Contact Number: ' || customer\_record.CONTACT\_NUMBER);

DBMS\_OUTPUT.PUT\_LINE('Email: ' || customer\_record.EMAIL);

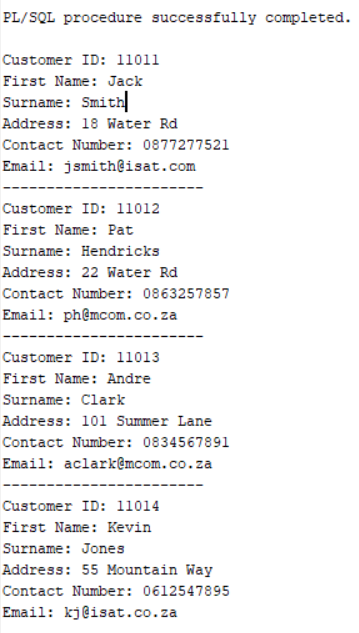
DBMS\_OUTPUT.PUT\_LINE('-----------------------');

END LOOP;

CLOSE customer\_cursor;

END;

/



## QUESTION 7.3

-- QUESTION 7.3(TutorialsPoint, n.d.)

SET SERVEROUTPUT ON;

-- PL/SQL block to define a custom exception

-- This block verifies if the donation price surpasses a certain limit

-- If it does, a custom exception is triggered

DECLARE

-- Define an exception for exceeding the allowed donation price

price\_limit\_exceeded EXCEPTION;

-- Define a variable to store the donation price

v\_price NUMBER;

-- Maximum permitted donation price

max\_price CONSTANT NUMBER := 2000; -- Example limit set to 1000

BEGIN

-- Fetch the donation price from the DONATION table

SELECT PRICE INTO v\_price

FROM DONATION

WHERE DONATION\_ID = 7117; -- Assuming donation ID 1 for illustration

-- Verify if the price goes beyond the limit

IF v\_price > max\_price THEN

-- Trigger the custom exception if the price is above the limit

RAISE price\_limit\_exceeded;

END IF;

-- If no exception is raised, insert data into the INVOICE table

INSERT INTO INVOICE (INVOICE\_NUM, CUSTOMER\_ID, INVOICE\_DATE, EMPLOYEE\_ID, DONATION\_ID, DELIVERY\_ID)

VALUES (8117, 11011, SYSDATE, 'emp103', 7117, 517);

DBMS\_OUTPUT.PUT\_LINE('Invoice successfully created.');

EXCEPTION

-- Handle the custom exception if the donation price is too high

WHEN price\_limit\_exceeded THEN

DBMS\_OUTPUT.PUT\_LINE('Error: Donation price exceeds the allowed maximum of ' || max\_price);

-- Handle any other potential exceptions

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Error: ' || SQLERRM);

END;

/

A white background with black text

Description automatically generated

# QUESTION 8

-- QUESTION 8

SELECT

c.FIRST\_NAME AS "First Name",

c.SURNAME AS "Surname",

SUM(d.PRICE) AS "Amount",

CASE

WHEN SUM(d.PRICE) >= 1500 THEN '\*\*\*'

WHEN SUM(d.PRICE) BETWEEN 1000 AND 1400 THEN '\*\*'

ELSE '\*'

6

END AS "Rating"

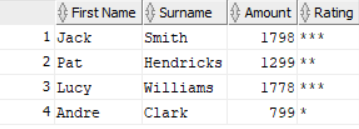
FROM CUSTOMER c

JOIN INVOICE i ON c.CUSTOMER\_ID = i.CUSTOMER\_ID

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

WHERE c.FIRST\_NAME IN ('Jack', 'Lucy', 'Pat', 'Andre')

GROUP BY c.FIRST\_NAME, c.SURNAME;



# REFERENCES

L. Jayapalan, Belden, E., Huey, P., Moore, S. and Rich, K. (2020). *%ROWTYPE Attribute*. [online] Oracle Help Center. Available at: https://docs.oracle.com/en/database/oracle/oracle-database/21/lnpls/ROWTYPE-attribute.html.

Moore, S., L. Jayapalan, Huey, P., Belden, E., Rich, K. and Moore, V. (2019). *%TYPE Attribute*. [online] Oracle Help Center. Available at: https://docs.oracle.com/en/database/oracle/oracle-database/19/lnpls/TYPE-attribute.html.

Oracle (2019). *CREATE SEQUENCE*. [online] Oracle Help Center. Available at: https://docs.oracle.com/en/database/oracle/oracle-database/19/sqlrf/CREATE-SEQUENCE.html.

The IIE (2024). *IIE Module Manual ADDB7311/ADDB6311 ADVANCED DATABASES MODULE MANUAL 2024 (First Edition: 2024)*.

TutorialsPoint (n.d.). *PL/SQL - Exceptions - Tutorialspoint*. [online] www.tutorialspoint.com. Available at: https://www.tutorialspoint.com/plsql/plsql\_exceptions.htm.