Name: Celebrate

Surname: Mashaba

Student number: ST10055395

Module name: ADVANCED DATABASES

Module code: ADDB6311

Assessment Type: Assignment 2

Group: 2

--Create CUSTOMER table

CREATE TABLE CUSTOMER (

CUSTOMER\_ID VARCHAR(10) PRIMARY KEY,

FIRST\_NAME VARCHAR(50),

SURNAME VARCHAR(50),

ADDRESS VARCHAR(100),

CONTACT\_NUMBER VARCHAR(15),

EMAIL VARCHAR(100)

);

CREATE TABLE EMPLOYEE (

EMPLOYEE\_ID VARCHAR(10) PRIMARY KEY,

FIRST\_NAME VARCHAR(50),

SURNAME VARCHAR(50),

CONTACT\_NUMBER VARCHAR(15),

ADDRESS VARCHAR(100),

EMAIL VARCHAR(100)

);

-- Create DONATOR table

CREATE TABLE DONATOR (

DONATOR\_ID VARCHAR(10) PRIMARY KEY,

FIRST\_NAME VARCHAR(50),

SURNAME VARCHAR(50),

CONTACT\_NUMBER VARCHAR(15),

EMAIL VARCHAR(100)

);

-- Create DONATION table

CREATE TABLE DONATION (

DONATION\_ID VARCHAR(10) PRIMARY KEY,

DONATOR\_ID VARCHAR(10),

DONATION VARCHAR(100),

PRICE VARCHAR(10),

DONATION\_DATE DATE,

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

);

-- Create DELIVERY table

CREATE TABLE DELIVERY (

DELIVERY\_ID VARCHAR(10) PRIMARY KEY,

DELIVERY\_NOTES VARCHAR(100),

DISPATCH\_DATE DATE,

DELIVERY\_DATE DATE

);

-- Create RETURNS table

CREATE TABLE RETURNS (

RETURN\_ID VARCHAR(10) PRIMARY KEY,

RETURN\_DATE DATE,

REASON VARCHAR(100),

CUSTOMER\_ID VARCHAR(10),

DONATION\_ID VARCHAR(10),

EMPLOYEE\_ID VARCHAR(10),

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

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

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

);

-- Create INVOICE table

CREATE TABLE INVOICE (

INVOICE\_NUM VARCHAR(10) PRIMARY KEY,

CUSTOMER\_ID VARCHAR(10),

INVOICE\_DATE DATE,

EMPLOYEE\_ID VARCHAR(10),

DONATION\_ID VARCHAR(10),

DELIVERY\_ID VARCHAR(10),

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

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

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

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

);

-- Populate CUSTOMER table

INSERT ALL

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

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

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

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

INTO CUSTOMER VALUES ('11015', 'Lucy', 'Williams', '5 Main rd', '0827238521', 'lw@mcal.co.za')

SELECT \* FROM DUAL;

-- Populate EMPLOYEE table

INSERT ALL

INTO EMPLOYEE VALUES ('emp101', 'Jeff', 'Davis', '0877277521', '10 main road', 'jand@isat.com')

INTO EMPLOYEE VALUES ('emp102', 'Kevin', 'Marks', '0837377522', '18 water road', 'km@isat.com')

INTO EMPLOYEE VALUES ('emp103', 'Adanya', 'Andrews', '0817117523', '21 circle lane', 'aa@isat.com')

INTO EMPLOYEE VALUES ('emp104', 'Adebayo', 'Dryer', '0797215244', '1 sea road', 'aryer@isat.com')

INTO EMPLOYEE VALUES ('emp105', 'Xolani', 'Samson', '0827122255', '12 main road', 'xosam@isat.com')

SELECT \* FROM DUAL;

-- Populate DONATOR table

INSERT ALL

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

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

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

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

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

SELECT \* FROM DUAL;

-- Populate DONATION table

INSERT ALL

INTO DONATION VALUES ('7111', '20111', 'KIC Fridge', 'R 599', TO\_DATE('2024-05-01', 'YYYY-MM-DD'))

INTO DONATION VALUES ('7112', '20112', 'Samsung 42inch LCD', 'R 1 299', TO\_DATE('2024-05-03', 'YYYY-MM-DD'))

INTO DONATION VALUES ('7113', '20113', 'Sharp Microwave', 'R 1 599', TO\_DATE('2024-05-03', 'YYYY-MM-DD'))

INTO DONATION VALUES ('7114', '20115', '6 Seat Dining room table', 'R 799', TO\_DATE('2024-05-05', 'YYYY-MM-DD'))

INTO DONATION VALUES ('7115', '20114', 'Lazyboy Sofa', 'R 1 199', TO\_DATE('2024-05-07', 'YYYY-MM-DD'))

INTO DONATION VALUES ('7116', '20113', 'JVC Surround Sound System', 'R 179', TO\_DATE('2024-05-09', 'YYYY-MM-DD'))

SELECT \* FROM DUAL;

-- Populate DELIVERY table

INSERT ALL

INTO DELIVERY VALUES ('511', 'Double packaging requested', TO\_DATE('2024-05-10', 'YYYY-MM-DD'), TO\_DATE('2024-05-15', 'YYYY-MM-DD'))

INTO DELIVERY VALUES ('512', 'Delivery to work address', TO\_DATE('2024-05-12', 'YYYY-MM-DD'), TO\_DATE('2024-05-15', 'YYYY-MM-DD'))

INTO DELIVERY VALUES ('513', 'Signature required', TO\_DATE('2024-05-12', 'YYYY-MM-DD'), TO\_DATE('2024-05-17', 'YYYY-MM-DD'))

INTO DELIVERY VALUES ('514', 'No notes', TO\_DATE('2024-05-12', 'YYYY-MM-DD'), TO\_DATE('2024-05-15', 'YYYY-MM-DD'))

INTO DELIVERY VALUES ('515', 'Birthday present wrapping required', TO\_DATE('2024-05-18', 'YYYY-MM-DD'), TO\_DATE('2024-05-19', 'YYYY-MM-DD'))

INTO DELIVERY VALUES ('516', 'Delivery to work address', TO\_DATE('2024-05-20', 'YYYY-MM-DD'), TO\_DATE('2024-05-25', 'YYYY-MM-DD'))

SELECT \* FROM DUAL;

-- Populate RETURNS table

INSERT ALL

INTO RETURNS VALUES ('ret001', TO\_DATE('2024-05-25', 'YYYY-MM-DD'), 'Customer not satisfied with product', '11011', '7116', 'emp101')

INTO RETURNS VALUES ('ret002', TO\_DATE('2024-05-25', 'YYYY-MM-DD'), 'Product had broken section', '11013', '7114', 'emp103')

SELECT \* FROM DUAL;

-- Populate INVOICE table

INSERT ALL

INTO INVOICE VALUES ('8111', '11011', TO\_DATE('2024-05-15', 'YYYY-MM-DD'), 'emp103', '7111', '511')

INTO INVOICE VALUES ('8112', '11013', TO\_DATE('2024-05-15', 'YYYY-MM-DD'), 'emp101', '7114', '512')

INTO INVOICE VALUES ('8113', '11012', TO\_DATE('2024-05-17', 'YYYY-MM-DD'), 'emp101', '7112', '513')

INTO INVOICE VALUES ('8114', '11015', TO\_DATE('2024-05-17', 'YYYY-MM-DD'), 'emp102', '7113', '514')

INTO INVOICE VALUES ('8115', '11011', TO\_DATE('2024-05-17', 'YYYY-MM-DD'), 'emp102', '7115', '515')

INTO INVOICE VALUES ('8116', '11015', TO\_DATE('2024-05-18', 'YYYY-MM-DD'), 'emp103', '7116', '516')

SELECT \* FROM DUAL;

SELECT \* FROM CUSTOMER;

SELECT \* FROM EMPLOYEE;

SELECT \* FROM DONATOR;

SELECT \* FROM DONATION;

SELECT \* FROM DELIVERY;

SELECT \* FROM RETURNS;

SELECT \* FROM INVOICE;

Question 2

--QUESTION2--

SELECT

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

e.EMPLOYEE\_ID AS Employee\_ID,

d.DELIVERY\_NOTES,

dn.DONATION,

i.INVOICE\_NUM,

i.INVOICE\_DATE

FROM

CUSTOMER c

JOIN

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

JOIN

DELIVERY d ON i.DELIVERY\_ID = d.DELIVERY\_ID

JOIN

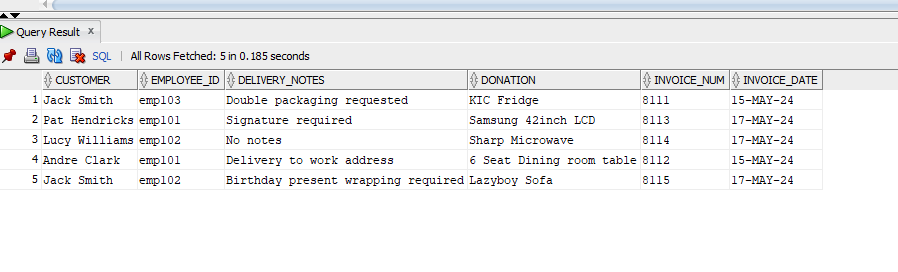
DONATION dn ON i.DONATION\_ID = dn.DONATION\_ID

JOIN

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

WHERE

i.INVOICE\_DATE < TO\_DATE('2024-05-18', 'YYYY-MM-DD');



Question 3

--QUESTION3--

-- Step 1: Create a sequence

CREATE SEQUENCE funding\_seq

START WITH 1

INCREMENT BY 1

NOCACHE

NOCYCLE;

-- Step 2: Create the Funding table

CREATE TABLE Funding (

funding\_id NUMBER PRIMARY KEY,

funder VARCHAR(100),

funding\_amount NUMBER

);

-- Step 3: Insert records into the Funding table

INSERT INTO Funding (funding\_id, funder, funding\_amount)

VALUES (funding\_seq.NEXTVAL, 'XYZ Foundation', 10000);

INSERT INTO Funding (funding\_id, funder, funding\_amount)

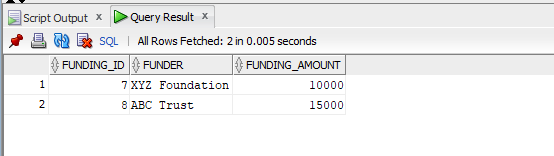
VALUES (funding\_seq.NEXTVAL, 'ABC Trust', 15000);

-- I create a sequence named "funding\_seq" to generate unique IDs for the "funding\_id" column.

-- The "Funding" table is created with columns for "funding\_id", "funder", and "funding\_amount".

-- When inserting records into the "Funding" table, I use "funding\_seq.NEXTVAL" to automatically generate a unique ID for the "funding\_id" column.

SELECT \* FROM Funding;



Question 4

--QUESTION4--

SET SERVEROUTPUT ON;

DECLARE

v\_customer\_name VARCHAR2(100);

v\_donation\_purchased VARCHAR2(100);

v\_donation\_price VARCHAR2(50);

v\_return\_reason VARCHAR2(100);

BEGIN

-- Open cursor to fetch data

FOR cur IN (

SELECT

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

dn.DONATION AS Donation\_Purchased,

dn.PRICE AS Donation\_Price,

r.REASON AS Return\_Reason

FROM

RETURNS r

JOIN

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

JOIN

DONATION dn ON r.DONATION\_ID = dn.DONATION\_ID

) LOOP

-- Fetch data from cursor

v\_customer\_name := cur.Customer\_Name;

v\_donation\_purchased := cur.Donation\_Purchased;

v\_donation\_price := cur.Donation\_Price;

v\_return\_reason := cur.Return\_Reason;

-- Display data

DBMS\_OUTPUT.PUT\_LINE('Customer Name: ' || v\_customer\_name);

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

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

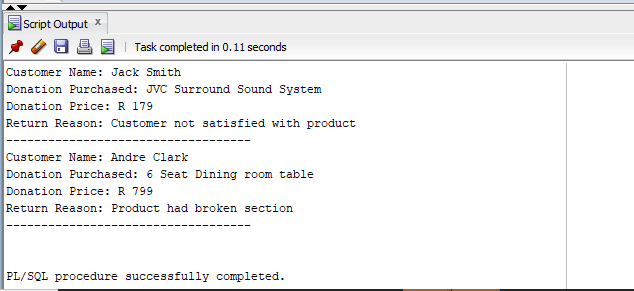
DBMS\_OUTPUT.PUT\_LINE('Return Reason: ' || v\_return\_reason);

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

END LOOP;

END;

/



Question 5

--QUESTION5--

SET SERVEROUTPUT ON;

DECLARE

v\_customer\_name VARCHAR2(100);

v\_employee\_name VARCHAR2(100);

v\_donation VARCHAR2(100);

v\_dispatch\_date DATE;

v\_delivery\_date DATE;

v\_days\_diff NUMBER;

BEGIN

-- Open cursor to fetch data

FOR cur IN (

SELECT

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

e.FIRST\_NAME || ' ' || e.SURNAME AS Employee\_Name,

dn.DONATION AS Donation,

del.DISPATCH\_DATE AS Dispatch\_Date,

del.DELIVERY\_DATE AS Delivery\_Date

FROM

INVOICE inv

JOIN

CUSTOMER c ON inv.CUSTOMER\_ID = c.CUSTOMER\_ID

JOIN

EMPLOYEE e ON inv.EMPLOYEE\_ID = e.EMPLOYEE\_ID

JOIN

DELIVERY del ON inv.DELIVERY\_ID = del.DELIVERY\_ID

JOIN

DONATION dn ON inv.DONATION\_ID = dn.DONATION\_ID

WHERE

c.CUSTOMER\_ID = '11013'

) LOOP

-- Fetch data from cursor

v\_customer\_name := cur.Customer\_Name;

v\_employee\_name := cur.Employee\_Name;

v\_donation := cur.Donation;

v\_dispatch\_date := cur.Dispatch\_Date;

v\_delivery\_date := cur.Delivery\_Date;

v\_days\_diff := v\_delivery\_date - v\_dispatch\_date; -- Calculate days difference

-- Display data

DBMS\_OUTPUT.PUT\_LINE('Customer Name: ' || v\_customer\_name);

DBMS\_OUTPUT.PUT\_LINE('Employee Name: ' || v\_employee\_name);

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

DBMS\_OUTPUT.PUT\_LINE('Dispatch Date: ' || TO\_CHAR(v\_dispatch\_date, 'DD Month YYYY'));

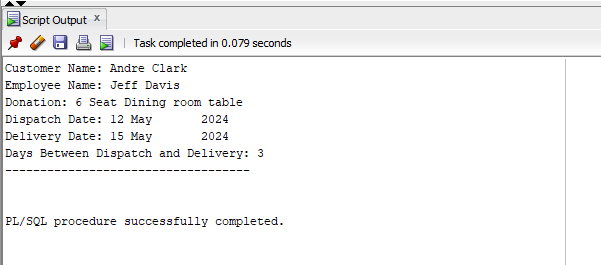
DBMS\_OUTPUT.PUT\_LINE('Delivery Date: ' || TO\_CHAR(v\_delivery\_date, 'DD Month YYYY'));

DBMS\_OUTPUT.PUT\_LINE('Days Between Dispatch and Delivery: ' || v\_days\_diff);

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

END LOOP;

END;



Question 6

--QUESTION6--

SET SERVEROUTPUT ON;

DECLARE

v\_customer\_name VARCHAR2(100);

v\_total\_amount\_spent VARCHAR2(100);

BEGIN

FOR cur IN (

SELECT

c.FIRST\_NAME AS First\_Name,

c.SURNAME AS Surname,

'R ' || TO\_CHAR(SUM(TO\_NUMBER(REGEXP\_REPLACE(dn.PRICE, '[^0-9]', '')))) AS Amount,

CASE

WHEN SUM(TO\_NUMBER(REGEXP\_REPLACE(dn.PRICE, '[^0-9]', ''))) >= 1500 THEN '(\*)'

ELSE ''

END AS Star\_Rating

FROM

CUSTOMER c

JOIN

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

JOIN

DONATION dn ON r.DONATION\_ID = dn.DONATION\_ID

GROUP BY

c.CUSTOMER\_ID, c.FIRST\_NAME, c.SURNAME

ORDER BY

c.CUSTOMER\_ID

) LOOP

v\_customer\_name := cur.First\_Name || ' ' || cur.Surname;

v\_total\_amount\_spent := cur.Amount || ' ' || cur.Star\_Rating;

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

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

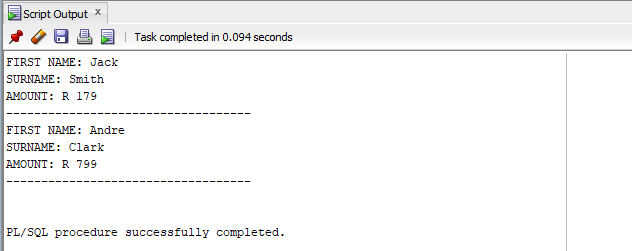
DBMS\_OUTPUT.PUT\_LINE('AMOUNT: ' || v\_total\_amount\_spent); -- Corrected output

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

END LOOP;

END;

/



Question 7.1

--QUESTION7--

-- Q.7.1 %TYPE attribute:

-- Define a variable of the same type as the 'CUSTOMER\_ID' column in the CUSTOMER table

DECLARE

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

BEGIN

-- Assign a value to the variable

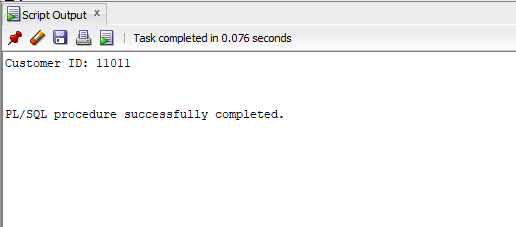
v\_cust\_id := '11011';

-- Display the value

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

END;

/



Q.7.2

-- Q.7.2 %ROWTYPE attribute:

-- Define a record variable based on the structure of the CUSTOMER table

DECLARE

v\_cust\_rec CUSTOMER%ROWTYPE;

BEGIN

-- Assign values to the fields of the record

v\_cust\_rec.CUSTOMER\_ID := '11011';

v\_cust\_rec.FIRST\_NAME := 'Jack';

v\_cust\_rec.SURNAME := 'Smith';

-- Display the values

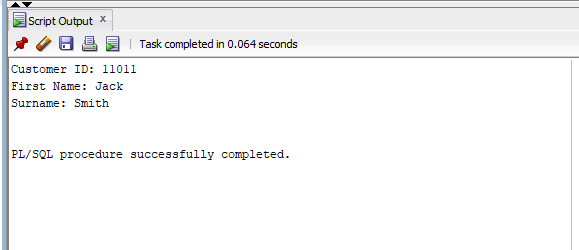
DBMS\_OUTPUT.PUT\_LINE('Customer ID: ' || v\_cust\_rec.CUSTOMER\_ID);

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

DBMS\_OUTPUT.PUT\_LINE('Surname: ' || v\_cust\_rec.SURNAME);

END;

/



Q7.3

-- Q.7.3 System defined exception:

DECLARE

v\_amount NUMBER := 1000;

v\_balance NUMBER := 500;

BEGIN

-- Check if the balance is sufficient

IF v\_amount > v\_balance THEN

-- Raise an exception if the balance is insufficient

RAISE VALUE\_ERROR;

ELSE

-- Process the transaction if the balance is sufficient

DBMS\_OUTPUT.PUT\_LINE('Transaction processed successfully.');

END IF;

EXCEPTION

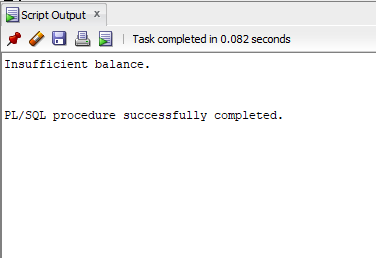
-- Catch the exception and handle it

WHEN VALUE\_ERROR THEN

DBMS\_OUTPUT.PUT\_LINE('Insufficient balance.');

END;

/



Q7.4

-- Q.7.4 User defined exception:

-- Define a user-defined exception

DECLARE

insufficient\_balance EXCEPTION;

v\_amount NUMBER := 1000;

v\_balance NUMBER := 500;

BEGIN

-- Check if the balance is sufficient

IF v\_amount > v\_balance THEN

-- Raise the user-defined exception if the balance is insufficient

RAISE insufficient\_balance;

ELSE

-- Process the transaction if the balance is sufficient

DBMS\_OUTPUT.PUT\_LINE('Transaction processed successfully.');

END IF;

EXCEPTION

-- Catch the user-defined exception and handle it

WHEN insufficient\_balance THEN

DBMS\_OUTPUT.PUT\_LINE('Insufficient balance.');

END;

/

