ST10021259

Assignment 2

INSY7213

Script:

-- CUSTOMER TABLE

CREATE TABLE CUSTOMER (

CUSTOMER\_ID NUMBER PRIMARY KEY,

FIRST\_NAME VARCHAR2(50) NOT NULL,

SURNAME VARCHAR2(50) NOT NULL,

ADDRESS VARCHAR2(100),

CONTACT\_NUMBER VARCHAR2(20),

EMAIL VARCHAR2(100)

);

-- EMPLOYEE TABLE

CREATE TABLE EMPLOYEE (

EMPLOYEE\_ID VARCHAR2(10) PRIMARY KEY,

FIRST\_NAME VARCHAR2(50) NOT NULL,

SURNAME VARCHAR2(50) NOT NULL,

CONTACT\_NUMBER VARCHAR2(20),

ADDRESS VARCHAR2(100),

EMAIL VARCHAR2(100)

);

-- DONATOR TABLE

CREATE TABLE DONATOR (

DONATOR\_ID NUMBER PRIMARY KEY,

FIRST\_NAME VARCHAR2(50) NOT NULL,

SURNAME VARCHAR2(50) NOT NULL,

CONTACT\_NUMBER VARCHAR2(20),

EMAIL VARCHAR2(100)

);

-- DONATION TABLE

CREATE TABLE DONATION (

DONATION\_ID NUMBER PRIMARY KEY,

DONATOR\_ID NUMBER NOT NULL,

DONATION VARCHAR(50),

DONATION\_PRICE NUMBER(10,2),

DONATION\_DATE DATE,

CONSTRAINT FK\_DONATOR FOREIGN KEY (DONATOR\_ID)

REFERENCES DONATOR(DONATOR\_ID)

);

-- DELIVERY TABLE

CREATE TABLE DELIVERY (

DELIVERY\_ID NUMBER PRIMARY KEY,

DELIVERY\_NOTES VARCHAR2(255),

DISPATCH\_DATE DATE,

DELIVERY\_DATE DATE

);

-- RETURNS TABLE

CREATE TABLE RETURNS (

RETURN\_ID VARCHAR2(10) PRIMARY KEY,

RETURN\_DATE DATE,

REASON VARCHAR2(255),

CUSTOMER\_ID NUMBER,

DONATION\_ID NUMBER,

EMPLOYEE\_ID VARCHAR2(10),

CONSTRAINT FK\_RETURN\_CUSTOMER FOREIGN KEY (CUSTOMER\_ID)

REFERENCES CUSTOMER(CUSTOMER\_ID),

CONSTRAINT FK\_RETURN\_DONATION FOREIGN KEY (DONATION\_ID)

REFERENCES DONATION(DONATION\_ID),

CONSTRAINT FK\_RETURN\_EMPLOYEE FOREIGN KEY (EMPLOYEE\_ID)

REFERENCES EMPLOYEE(EMPLOYEE\_ID)

);

-- INVOICE TABLE

CREATE TABLE INVOICE (

INVOICE\_NUM NUMBER PRIMARY KEY,

CUSTOMER\_ID NUMBER,

INVOICE\_DATE DATE,

EMPLOYEE\_ID VARCHAR2(10),

DONATION\_ID NUMBER,

DELIVERY\_ID NUMBER,

CONSTRAINT FK\_INVOICE\_CUSTOMER FOREIGN KEY (CUSTOMER\_ID)

REFERENCES CUSTOMER(CUSTOMER\_ID),

CONSTRAINT FK\_INVOICE\_EMPLOYEE FOREIGN KEY (EMPLOYEE\_ID)

REFERENCES EMPLOYEE(EMPLOYEE\_ID),

CONSTRAINT FK\_INVOICE\_DONATION FOREIGN KEY (DONATION\_ID)

REFERENCES DONATION(DONATION\_ID),

CONSTRAINT FK\_INVOICE\_DELIVERY FOREIGN KEY (DELIVERY\_ID)

REFERENCES DELIVERY(DELIVERY\_ID)

);

-- INSERT SAMPLE DATA (MATCHES TABLE STRUCTURE)

-- 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', 'Hendrix', '22 Water Road', '0863257857', 'ph@mcom.co.za')

INTO CUSTOMER (CUSTOMER\_ID, FIRST\_NAME, SURNAME, ADDRESS, CONTACT\_NUMBER, EMAIL)

VALUES (11013, 'Andre', 'Clark', '101 Summer Lane', '08345767891', '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 Road', '0827238521', 'lw@mcal.co.za')

SELECT \* FROM dual;

-- EMPLOYEE

INSERT ALL

INTO EMPLOYEE (EMPLOYEE\_ID, FIRST\_NAME, SURNAME, CONTACT\_NUMBER, ADDRESS, EMAIL)

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

INTO EMPLOYEE (EMPLOYEE\_ID, FIRST\_NAME, SURNAME, CONTACT\_NUMBER, ADDRESS, EMAIL)

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

INTO EMPLOYEE (EMPLOYEE\_ID, FIRST\_NAME, SURNAME, CONTACT\_NUMBER, ADDRESS, EMAIL)

VALUES ('emp103', 'Adania', 'Andrews', '081711-7523', '21 Circle Lane', 'aa@isat.com')

INTO EMPLOYEE (EMPLOYEE\_ID, FIRST\_NAME, SURNAME, CONTACT\_NUMBER, ADDRESS, EMAIL)

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

INTO EMPLOYEE (EMPLOYEE\_ID, FIRST\_NAME, SURNAME, CONTACT\_NUMBER, ADDRESS, EMAIL)

VALUES ('emp105', 'Xolani', 'Samson', '082712-2255', '12 Main Road', 'xosam@isat.com')

SELECT \* FROM dual;

-- 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, 'Stephan', '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', '8026575650', 'kross@gsat.com')

INTO DONATOR (DONATOR\_ID, FIRST\_NAME, SURNAME, CONTACT\_NUMBER, EMAIL)

VALUES (20115, 'Abraham', 'Clark', '8797656430', 'aclark@ymail.com')

SELECT \* FROM dual;

-- DONATION

INSERT ALL

INTO DONATION (DONATION\_ID, DONATOR\_ID, DONATION, DONATION\_PRICE, DONATION\_DATE)

VALUES (7111, 20111, 'KIC Fridge', 599.00, TO\_DATE('01-05-2024', 'DD-MM-YYYY'))

INTO DONATION (DONATION\_ID, DONATOR\_ID, DONATION, DONATION\_PRICE, DONATION\_DATE)

VALUES (7112, 20112, 'Samsung 42-inch LCD', 1299.00, TO\_DATE('03-05-2024', 'DD-MM-YYYY'))

INTO DONATION (DONATION\_ID, DONATOR\_ID, DONATION, DONATION\_PRICE, DONATION\_DATE)

VALUES (7113, 20113, 'Sharp Microwave', 1599.00, TO\_DATE('03-05-2024', 'DD-MM-YYYY'))

INTO DONATION (DONATION\_ID, DONATOR\_ID, DONATION, DONATION\_PRICE, DONATION\_DATE)

VALUES (7114, 20115, '6 seat dining room table', 799.00, TO\_DATE('05-05-2024', 'DD-MM-YYYY'))

INTO DONATION (DONATION\_ID, DONATOR\_ID, DONATION, DONATION\_PRICE, DONATION\_DATE)

VALUES (7115, 20114, 'Lazy Boy Sofa', 1199.00, TO\_DATE('07-05-2024', 'DD-MM-YYYY'))

INTO DONATION (DONATION\_ID, DONATOR\_ID, DONATION, DONATION\_PRICE, DONATION\_DATE)

VALUES (7116, 20113, 'JVC Surround Sound System', 179.00, TO\_DATE('09-05-2024', 'DD-MM-YYYY'))

SELECT \* FROM dual;

-- DELIVERY

INSERT ALL

INTO DELIVERY (DELIVERY\_ID, DELIVERY\_NOTES, DISPATCH\_DATE, DELIVERY\_DATE)

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

INTO DELIVERY (DELIVERY\_ID, DELIVERY\_NOTES, DISPATCH\_DATE, DELIVERY\_DATE)

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

INTO DELIVERY (DELIVERY\_ID, DELIVERY\_NOTES, DISPATCH\_DATE, DELIVERY\_DATE)

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

INTO DELIVERY (DELIVERY\_ID, DELIVERY\_NOTES, DISPATCH\_DATE, DELIVERY\_DATE)

VALUES (514, NULL, TO\_DATE('12-05-2024','DD-MM-YYYY'), TO\_DATE('15-05-2024','DD-MM-YYYY'))

INTO DELIVERY (DELIVERY\_ID, DELIVERY\_NOTES, DISPATCH\_DATE, DELIVERY\_DATE)

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

INTO DELIVERY (DELIVERY\_ID, DELIVERY\_NOTES, DISPATCH\_DATE, DELIVERY\_DATE)

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

SELECT \* FROM dual;

-- RETURNS

INSERT ALL

INTO RETURNS (RETURN\_ID, RETURN\_DATE, REASON, CUSTOMER\_ID, DONATION\_ID, EMPLOYEE\_ID)

VALUES ('ret001', TO\_DATE('25-05-2024','DD-MM-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('25-05-2024','DD-MM-YYYY'), 'Product had broken section', 11013, 7114, 'emp103')

SELECT \* FROM dual;

-- INVOICE TABLE - INSERT ALL SAMPLE DATA

INSERT ALL

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

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

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

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

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

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

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

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

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

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

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

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

SELECT \* FROM dual;

-- COMMIT CHANGES

COMMIT;

--QUESTION 2:

SELECT

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

i.EMPLOYEE\_ID,

d.DELIVERY\_NOTES,

dn.DONATION AS DONATION\_PURCHASED,

i.INVOICE\_NUM,

TO\_CHAR(i.INVOICE\_DATE, 'DD-MON-YYYY') AS INVOICE\_DATE

FROM

INVOICE i

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

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

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

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

WHERE

i.INVOICE\_DATE > TO\_DATE('16-05-2024', 'DD-MM-YYYY')

ORDER BY

i.INVOICE\_DATE;

--QUESTION 3:

-- 3.1. Create FUNDING table

CREATE TABLE FUNDING (

FUNDING\_ID NUMBER PRIMARY KEY,

FUNDER VARCHAR2(100),

FUNDING\_AMOUNT NUMBER(12,2)

);

-- 3.2. Create a sequence to auto-generate FUNDING\_ID

CREATE SEQUENCE FUNDING\_SEQ

START WITH 1

INCREMENT BY 1

NOCACHE

NOCYCLE;

--Create a trigger that will automatically set the FUNDING\_ID on insert

CREATE OR REPLACE TRIGGER FUNDING\_BI

BEFORE INSERT ON FUNDING

FOR EACH ROW

BEGIN

IF :NEW.FUNDING\_ID IS NULL THEN

:NEW.FUNDING\_ID := FUNDING\_SEQ.NEXTVAL;

END IF;

END;

/

-- 3.3. Example insert statement (no need to provide FUNDING\_ID manually)

INSERT INTO FUNDING (FUNDER, FUNDING\_AMOUNT)

VALUES ('Global Aid Foundation', 50000.00);

-- COMMENT / JUSTIFICATION

-- Making use of a sequence with a BEFORE INSERT trigger makes it so that FUNDING\_ID will be automatically generated

--QUESTION 4:

-- Display returns information using PL/SQL

SET SERVEROUTPUT ON;

DECLARE

v\_customer\_name VARCHAR2(100);

v\_donation VARCHAR2(50);

v\_price NUMBER(10,2);

v\_reason VARCHAR2(255);

CURSOR returns\_cursor IS

SELECT

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

d.DONATION,

d.DONATION\_PRICE,

r.REASON

FROM

RETURNS r

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

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

ORDER BY r.RETURN\_ID;

BEGIN

FOR rec IN returns\_cursor LOOP

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

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

DBMS\_OUTPUT.PUT\_LINE('Price: R' || TO\_CHAR(rec.DONATION\_PRICE, '9999.99'));

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

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

END LOOP;

END;

/

--QUESTION 5:

-- Display customer name, employee name, donation, dispatch and delivery dates with days between for customer 11011

SET SERVEROUTPUT ON;

DECLARE

v\_customer\_name VARCHAR2(100);

v\_employee\_name VARCHAR2(100);

v\_donation VARCHAR2(50);

v\_dispatch\_date DATE;

v\_delivery\_date DATE;

v\_days\_to\_delivery NUMBER;

CURSOR invoice\_cursor IS

SELECT

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

e.FIRST\_NAME || ' ' || e.SURNAME AS EMPLOYEE\_NAME,

dn.DONATION,

d.DISPATCH\_DATE,

d.DELIVERY\_DATE,

(d.DELIVERY\_DATE - d.DISPATCH\_DATE) AS DAYS\_TO\_DELIVERY

FROM

INVOICE i

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

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

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

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

WHERE

c.CUSTOMER\_ID = 11011;

BEGIN

FOR rec IN invoice\_cursor LOOP

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

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

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

DBMS\_OUTPUT.PUT\_LINE('Dispatch Date: ' || TO\_CHAR(rec.DISPATCH\_DATE, 'DD-MON-YYYY'));

DBMS\_OUTPUT.PUT\_LINE('Delivery Date: ' || TO\_CHAR(rec.DELIVERY\_DATE, 'DD-MON-YYYY'));

DBMS\_OUTPUT.PUT\_LINE('Days to Delivery: ' || rec.DAYS\_TO\_DELIVERY);

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

END LOOP;

END;

/

--QUESTION 6:

DECLARE

v\_customer\_name VARCHAR2(100);

v\_total\_amount NUMBER(10,2);

v\_rating VARCHAR2(20);

CURSOR customer\_spending\_cursor IS

SELECT

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

NVL(SUM(dn.DONATION\_PRICE), 0) AS TOTAL\_AMOUNT

FROM

CUSTOMER c

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

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

GROUP BY

c.FIRST\_NAME, c.SURNAME

ORDER BY

TOTAL\_AMOUNT DESC;

BEGIN

FOR rec IN customer\_spending\_cursor LOOP

v\_customer\_name := rec.CUSTOMER\_NAME;

v\_total\_amount := rec.TOTAL\_AMOUNT;

-- Determine rating

IF v\_total\_amount >= 1500 THEN

v\_rating := ' (3 stars)';

ELSE

v\_rating := '';

END IF;

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

DBMS\_OUTPUT.PUT\_LINE('Amount: R' || TO\_CHAR(v\_total\_amount, '9999.99') || v\_rating);

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

END LOOP;

DBMS\_OUTPUT.PUT\_LINE('PL/SQL procedure successfully completed.');

END;

/

--QUESTION 7:

--7.1 %TYPE Attribute

DECLARE

v\_donation\_price DONATION.DONATION\_PRICE%TYPE; -- Uses %TYPE to match column data type

v\_budget\_limit DONATION.DONATION\_PRICE%TYPE := 1000.00; -- Budget limit for comparison

v\_donation\_name DONATION.DONATION%TYPE; -- Stores donation name

BEGIN

-- Retrieve a specific donation's price and name

SELECT DONATION\_PRICE, DONATION

INTO v\_donation\_price, v\_donation\_name

FROM DONATION

WHERE DONATION\_ID = 7112;

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

DBMS\_OUTPUT.PUT\_LINE('Price: R' || TO\_CHAR(v\_donation\_price, '9999.99'));

-- Check if within budget

IF v\_donation\_price <= v\_budget\_limit THEN

DBMS\_OUTPUT.PUT\_LINE('Status: Within budget limit');

ELSE

DBMS\_OUTPUT.PUT\_LINE('Status: Exceeds budget limit by R' ||

TO\_CHAR(v\_donation\_price - v\_budget\_limit, '9999.99'));

END IF;

END;

/

--7.2 %ROWTYPE Attribute

DECLARE

v\_employee\_record EMPLOYEE%ROWTYPE;

v\_emp\_id EMPLOYEE.EMPLOYEE\_ID%TYPE := 'emp103';

BEGIN

SELECT \*

INTO v\_employee\_record

FROM EMPLOYEE

WHERE EMPLOYEE\_ID = v\_emp\_id;

-- Display all employee details

DBMS\_OUTPUT.PUT\_LINE('===== EMPLOYEE DETAILS =====');

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

DBMS\_OUTPUT.PUT\_LINE('Full Name: ' || v\_employee\_record.FIRST\_NAME || ' ' || v\_employee\_record.SURNAME);

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

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

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

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

END;

/

--7.3 User-Defined Exception

DECLARE

-- User-defined exception

e\_invalid\_delivery\_dates EXCEPTION;

v\_delivery\_id DELIVERY.DELIVERY\_ID%TYPE;

v\_dispatch\_date DELIVERY.DISPATCH\_DATE%TYPE;

v\_delivery\_date DELIVERY.DELIVERY\_DATE%TYPE;

v\_notes DELIVERY.DELIVERY\_NOTES%TYPE;

CURSOR delivery\_cursor IS

SELECT DELIVERY\_ID, DISPATCH\_DATE, DELIVERY\_DATE, DELIVERY\_NOTES

FROM DELIVERY;

BEGIN

DBMS\_OUTPUT.PUT\_LINE('===== DELIVERY VALIDATION CHECK =====');

FOR rec IN delivery\_cursor LOOP

v\_delivery\_id := rec.DELIVERY\_ID;

v\_dispatch\_date := rec.DISPATCH\_DATE;

v\_delivery\_date := rec.DELIVERY\_DATE;

v\_notes := rec.DELIVERY\_NOTES;

-- Check if dispatch date is after delivery date (invalid scenario)

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

RAISE e\_invalid\_delivery\_dates; -- Raise custom exception

END IF;

DBMS\_OUTPUT.PUT\_LINE('Delivery ID ' || v\_delivery\_id || ': Valid');

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

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

DBMS\_OUTPUT.PUT\_LINE(' Days taken: ' || (v\_delivery\_date - v\_dispatch\_date));

END LOOP;

DBMS\_OUTPUT.PUT\_LINE('All delivery dates validated successfully!');

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

EXCEPTION

WHEN e\_invalid\_delivery\_dates THEN

DBMS\_OUTPUT.PUT\_LINE('ERROR: Invalid delivery dates detected!');

DBMS\_OUTPUT.PUT\_LINE('Delivery ID ' || v\_delivery\_id || ' has dispatch date after delivery date.');

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

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

DBMS\_OUTPUT.PUT\_LINE('Please correct the dates in the database.');

WHEN OTHERS THEN

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

END;

/

--QUESTION 8:

SELECT

c.FIRST\_NAME,

c.SURNAME,

'R' || TO\_CHAR(NVL(SUM(dn.DONATION\_PRICE), 0), '9999.99') AS AMOUNT,

CASE

WHEN NVL(SUM(dn.DONATION\_PRICE), 0) >= 1500 THEN '\*\*\*'

WHEN NVL(SUM(dn.DONATION\_PRICE), 0) BETWEEN 1000 AND 1499.99 THEN '\*\*'

ELSE '\*'

END AS CUSTOMER\_RATING

FROM

CUSTOMER c

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

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

GROUP BY

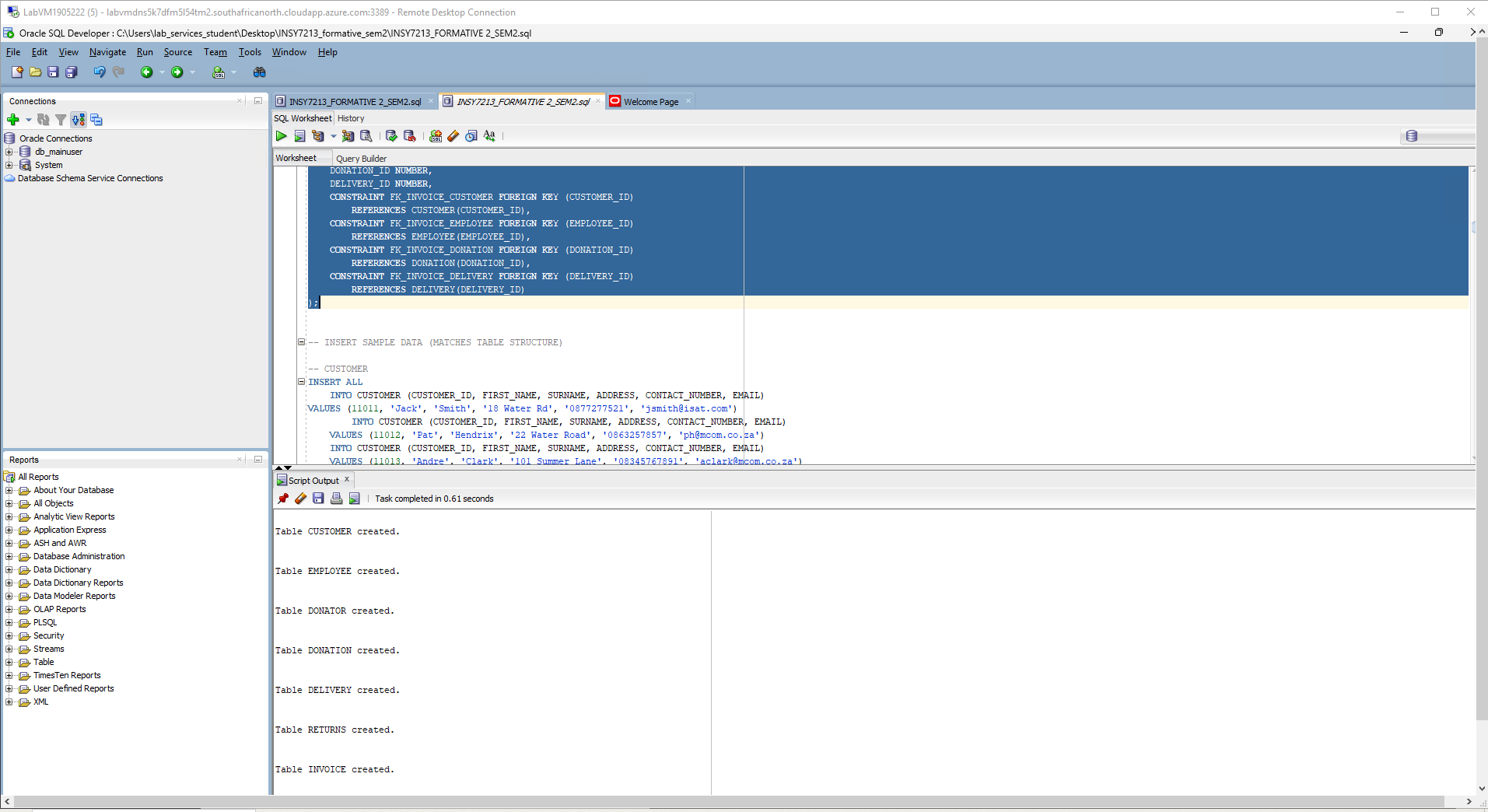
c.FIRST\_NAME, c.SURNAME

ORDER BY

NVL(SUM(dn.DONATION\_PRICE), 0) DESC;

Screenshots:

Create tables:

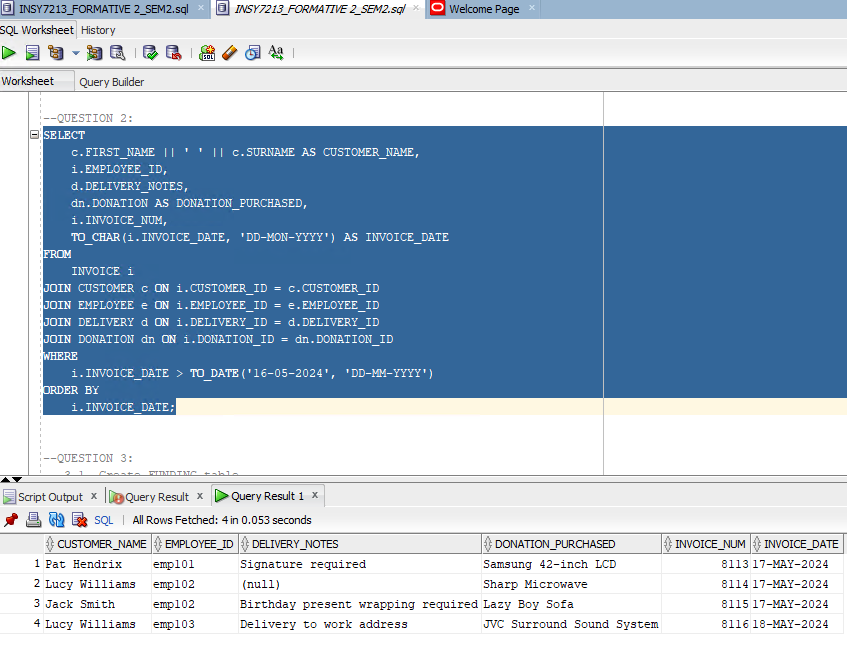


Insert data:

A screenshot of a computer

AI-generated content may be incorrect.

Question 2:

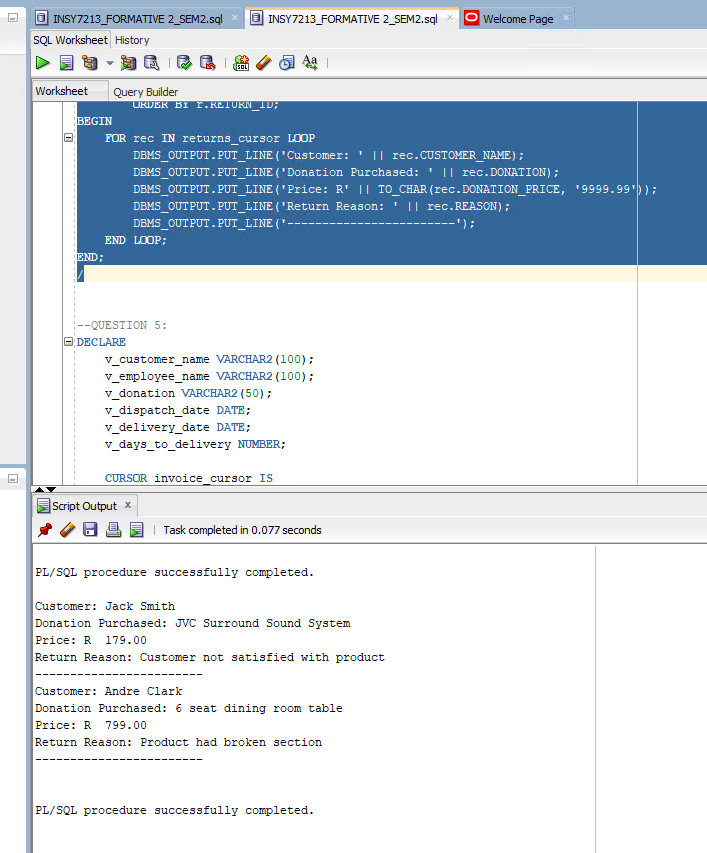


Question 3:

A screenshot of a computer

AI-generated content may be incorrect.

Question 4:



Question 5:

A screenshot of a computer

AI-generated content may be incorrect.

Question 6:

A screenshot of a computer

AI-generated content may be incorrect.

Question 7:

A screenshot of a computer

AI-generated content may be incorrect.

Question 8:

A screenshot of a computer

AI-generated content may be incorrect.