ADDB7311 ASSIGNMENT 2

Submission

Onello Travis Tarjanne

QUESTION 1

```
CREATE TABLE Customer (
  Customer_ID INT PRIMARY KEY,
  First_Name VARCHAR(50),
  Surname VARCHAR(50),
  Address VARCHAR(50),
  Contact_Number VARCHAR(15),
  Email VARCHAR(50)
);
-- Data insert for customer table
INSERT INTO Customer VALUES (11011, 'Jack', 'Smith', '18 Water Rd', '0877277521', 'jsmith@isat.com'),
(11012, 'Pat', 'Hendricks', '22 Water Rd', '0863257857', 'ph@mcom.co.za'),
(11013, 'Andre', 'Clark', '101 Summer Lane', '0834567891', 'aclark@mcom.co.za'),
(11014, 'Kevin', 'Jones', '5 Mountain way', '0612547895', 'kj@isat.co.za'),
(11015, 'Lucy', 'Williams', '5 Main rd', '0827238521', 'lw@mcal.co.za');

CREATE TABLE Employee (
  Employee_ID VARCHAR(50) PRIMARY KEY,
  First_Name VARCHAR(50),
  Surname VARCHAR(50),
  Contact_Number VARCHAR(15),
  Address VARCHAR(50),
  Email VARCHAR(50)
);
-- Data insert employee table
INSERT INTO Employee VALUES ('emp101', 'Jeff', 'Davis','0877277521','10 main road','jand@isat.com'),
('emp102', 'Kevin', 'Marks','0837377522','18 water road','km@isat.com'),
('emp103', 'Adanya', 'Andrews','0817117523','21 circle lane','aa@isat.com'),
('emp104', 'Abedayo', 'Dryer','0797215244','1 sea road','aryer@isat.com'),
('emp105', 'Xolani', 'Samson','0827122255','12 main road','xosam@isat.com');
CREATE TABLE Donator (
  Donator_ID INT PRIMARY KEY,
  First_Name VARCHAR(50),
  Surname VARCHAR(50),
  Contact_Number INT,
```

```
  Email VARCHAR(50)
);
-- Data insert for donator table
INSERT INTO Donator VALUES (20111, 'Jeff', 'Watson','0827172250', 'jwatson@ymail.com'),
(20112, 'Stephen', 'Jones','0837865670', 'joness@ymail.com');
(20113, 'James', 'Joe','0878978650', 'jj@isat.com'),
(20114, 'Kelly', 'Ross','0826575650', 'kross@gsat.com'),
(20115, 'Abraham', 'Clark','0797656430', 'aclark@ymail.com');
CREATE TABLE Donation (
  Donation_ID INT PRIMARY KEY,
  Donator_ID INT,
  Donation VARCHAR(50),
  Price DECIMAL(10, 2),
  Donation_Date DATE,
  FOREIGN KEY (Donator_ID) REFERENCES Donator(Donator_ID));
-- Data insert donation table
INSERT INTO Donation VALUES (7111,20111,'KIC fridge',599.00,'2024-05-1'),
(7112,20112,'Samsung 42 inch LCD',1299.00,'2024-05-3'),
(7113,20113,'Sharp Microwave',1599.00,'2024-05-3'),
(7114,20114,'6 seat Dinning room table ',799.00,'2024-05-5'),
(7115,20115,'Lazyboy Sofa',1199.00,'2024-05-7'),
(7116,20116,'JVC Surround Sound System',179.00,'2024-05-9');
CREATE TABLE Delivery (
  Delivery_ID INT PRIMARY KEY,
  Delivery_Notes VARCHAR(50),
  Dispatch_Date DATE,
  Delivery_Date DATE
);
-- Data insert for delivery table
INSERT INTO Delivery VALUES (511,'Double packaging requested','2024-05-10','2024-05-15'),
(512,'Delivery to work address','2024-05-12','2024-05-15'),
(513,'Signature required','2024-05-12','2024-05-17'),
(514,'No notes','2024-05-12','2024-05-15'),
(515,'Birthday present wrapping required','2024-05-18','2024-05-19'),
(516,'Delivery to work address','2024-05-20','2024-05-25');
```

```
CREATE TABLE Returns (
  Return_ID VARCHAR(50) PRIMARY KEY,
  Return_Date DATE,
  Reason VARCHAR(50),
  Customer_ID INT,
  Donation_ID INT,
  Employee_ID VARCHAR(50),
  FOREIGN KEY (Customer_ID) REFERENCES Customer(Customer_ID),
  FOREIGN KEY (Donation_ID) REFERENCES Donation(Donation_ID),
  FOREIGN KEY (Employee_ID) REFERENCES Employee(Employee_ID));
-- Data insert for Returns table
INSERT INTO Returns VALUES ('ret001','2024-05-25','Customer not satisfied with product',11011,7116,'emp101'),
('ret002','2024-05-25','Product had broken section',11013,7114,'emp103');
CREATE TABLE Invoice (
  Invoice_num INT PRIMARY KEY,
  Customer_ID INT,
  Invoice_Date DATE,
  Employee_ID VARCHAR(50),
  Donation_ID INT,
  Delivery_ID INT,
  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)
);
-- Data insert for Invoice table
INSERT INTO Invoice VALUES  (8111,11011,'2024-05-15','emp103',7111,511),
(8112,11013,'2024-05-15','emp101',7114,512),
(8113,11012,'2024-05-17','emp101',7112,513),
(8114,11015,'2024-05-17','emp102',7113,514),
(8115,11011,'2024-05-17','emp102',7115,515),
(8116,11015,'2024-05-18','emp1103',7116,516);
```

QUESTION 2

```
SELECT
   c.First_Name AS Customer_First_Name,
   c.Surname AS Customer_Surname,
   e.First_Name AS Employee_First_Name,
   e.Surname AS Employee_Surname,
   d.First_Name AS Donator_First_Name,
   d.Surname AS Donator_Surname,
   don.Donation,
   don.Price,
   del.Delivery_Notes,
   i.Invoice_num,
   i.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
   Donation don ON i.Donation_ID = don.Donation_ID
JOIN
   Donator d ON don.Donator_ID = d.Donator_ID
JOIN
   Delivery del ON i.Delivery_ID = del.Delivery_ID
WHERE
   i.invoice_date > '2024-05-16';
```

QUESTION 3

```
CREATE TABLE Funding(
   Fund_ID INT IDENTITY(1,1) PRIMARY KEY,
   Funder VARCHAR(50),
   Funding_Amount Decimal
);
```

```
-- Data inset for funding table

INSERT INTO Funding (Funder, Funding_Amount) VALUES ('John Baptist', 1212.88);
```

The Identity provided in the solution will give an id based on the number of the insert into the table ,giving each entry a unique ID. The insert is used to insert the values into those specific columns as the Fund_ID will be created automatically.

QUESTION 4

```
SELECT
  c.First_Name AS CUSTOMER,
  d.Donation AS DONATION_PURCHASED,
  d.Price AS PRICE,
  r.Reason AS RETURN_REASON
FROM
  Customers c
JOIN
  Donations d ON c.Customer_ID = d.Customer_ID
JOIN
  Returns r ON d.Donation_ID = r.Donation_ID;
```

QUESTION 5

```
SELECT
  c.First_Name AS CUSTOMER,
  e.First_Name AS EMPLOYEE,
  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_DELIVERY
FROM
  Customers c
JOIN
  Donations d ON c.Customer_ID = d.Customer_ID
JOIN
  Employees e ON d.Employee_ID = e.Employee_ID
JOIN
  Deliveries del ON d.Donation_ID = del.Donation_ID;
```

QUESTION 6

```
SELECT
  c.First_Name,
  c.Surname,
  CONCAT(SUM(don.Price), CASE
    WHEN SUM(don.Price) > 1500 THEN ' (*** )'
    ELSE ''
  END) AS Amount
FROM
  Customer c
JOIN
  Invoice i ON c.Customer_ID = i.Customer_ID
JOIN
  Donation don ON i.Donation_ID = don.Donation_ID
GROUP BY
  c.Customer_ID, c.First_Name, c.Surname
HAVING
  SUM(don.Price) > 0;
```

QUESTION 7

7.1) The %TYPE attribute is used to declare variables that match the data types of specific columns in the tables(IBM,2021).

Example(Searching for the name of an employee):

```
DECLARE
  v_employee_name Employee.employee_name%TYPE;  -- Matches the data type of employee_name
BEGIN
  SELECT employee_name INTO v_employee_name
  FROM Employee
  WHERE employee_id = 'emp102';
  DBMS_OUTPUT.PUT_LINE('Employee Name for ID emp102: ' || v_employee_name);
EXCEPTION
  WHEN NO_DATA_FOUND THEN
    DBMS_OUTPUT.PUT_LINE('No employee found with the ID emp102.');
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('An unexpected error occurred.');
```

```
END;
/
```

Output: 'Employee Name for ID emp102: Kevin

7.2) The %ROWTYPE attribute is useful when you want to fetch an entire row from a table(IBM,2021).

Example(Viewing the donation of a specific donor using donator id):

```
DECLARE
  v_donation Donation%ROWTYPE;  -- Record variable to hold an entire row from the Donation table
BEGIN
  SELECT * INTO v_donation
  FROM Donation
  WHERE Donator_ID = 7114;
  DBMS_OUTPUT.PUT_LINE('Donation: ' || v_donation.Donation);
  DBMS_OUTPUT.PUT_LINE('Price: ' || v_donation.Price);
  DBMS_OUTPUT.PUT_LINE('Donation Date: ' || v_donation.Donation_Date);
EXCEPTION
  WHEN NO_DATA_FOUND THEN
    DBMS_OUTPUT.PUT_LINE('No donation found for the specified Donator ID.');
  WHEN TOO_MANY_ROWS THEN
    DBMS_OUTPUT.PUT_LINE('Multiple donations found for the specified Donator ID.');
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('An unexpected error occurred.');
END;
/
```

Output:

Donation: ‘6 seating Dining room table’

Price:799.00

Donation Date: 2024-05-5

7.3) User-defined exceptions are used to handle any errors that may be made by the user, they are declared then raised explicitly(IIE,2024).

Example(Recording a donation):

```
DECLARE
  email_missing EXCEPTION;
  v_donator_id Donator.Donator_ID%TYPE;
```

```
  v_email Donator.Email%TYPE;
  v_donation VARCHAR(50);
  v_price DECIMAL(10, 2);
  v_donation_date DATE;
BEGIN
  v_donator_id := 20113;
  v_donation := 'KIC fridge';
  v_price := 599.00;
  v_donation_date := TO_DATE('2024-05-01', 'YYYY-MM-DD');
  SELECT Email INTO v_email FROM Donator WHERE Donator_ID = v_donator_id;
  IF v_email IS NULL THEN
    RAISE email_missing
  END IF;
  INSERT INTO Donation (Donation_ID, Donator_ID, Donation, Price, Donation_Date)
  VALUES (7117, v_donator_id, v_donation, v_price, v_donation_date);
  DBMS_OUTPUT.PUT_LINE('Donation recorded successfully');
EXCEPTION
  WHEN email_missing THEN
    DBMS_OUTPUT.PUT_LINE('Error: Email is required to enter a donation.');
  WHEN NO_DATA_FOUND THEN
    DBMS_OUTPUT.PUT_LINE('Error: Donor not found.');
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('An unexpected error occurred: ' || SQLERRM);
END;
```

Output:

Donation recorded successfully

QUESTION 8

```
SELECT
  c.First_Name,
  c.Surname,
  CONCAT(SUM(don.Price), CASE
    WHEN SUM(don.Price) > 1500 THEN ' (*** )'
    WHEN SUM(don.Price) BETWEEN 1000 AND 1400 THEN ' (** )'
    ELSE ' (*)'
  END) AS Amount
```

```
FROM
  Customer c
JOIN
  Invoice i ON c.Customer_ID = i.Customer_ID
JOIN
  Donation don ON i.Donation_ID = don.Donation_ID
GROUP BY
  c.Customer_ID, c.First_Name, c.Surname
HAVING
  SUM(don.Price) > 0;
```

REFERENCES

Ibm.com. (2021). *IBM Integrated Analytics System*. [online] Available at: https://www.ibm.com/docs/en/ias?topic=plsql-type-attribute-in-variable-declarations.

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

Ibm.com. (2021). *IBM Integrated Analytics System*. [online] Available at: https://www.ibm.com/docs/en/ias?topic=plsql-rowtype-attribute-in-record-type-declarations.

Gaetjen, S., Knox, D. and Maroulis, W. (2015) Oracle Database 12C security, O'Reilly Online Learning. Available at: https://www.oreilly.com/library/view/oracledatabase-12c/9780071824286/ (Accessed: 27 August 2024).

IIE(2024) ADVANCED DATABASES MODULE MANUAL Available at: https://advtechonline.sharepoint.com/sites/TertiaryStudents/IIE%20Student%20Materials/Forms/Default%20View.aspx?id=%2Fsites%2FTertiaryStudents%2FIIE%20Student%20Materials%2FNew%20Student%20Materials%20CAT%2FADDB7311%2F2024%2FADDB7311MM%2Epdf&viewid=db15e059%2D4f93%2D487f%2Dabda%2De538b821c7b8&parent=%2Fsites%2FTertiaryStudents%2FIIE%20Student%20Materials%2FNew%20Student%20Materials%20CAT%2FADDB7311%2F2024