

Summative Project

ADDB7311



November 7, 2024

ST10393280

Daniel Luke James

Table of Contents

[Question 1: 2](#_Toc181914700)

[Question 2: 12](#_Toc181914701)

[12](#_Toc181914702)

[Question 3: 13](#_Toc181914703)

[13](#_Toc181914704)

[Question 4: 14](#_Toc181914705)

[14](#_Toc181914706)

[Question 5: 15](#_Toc181914707)

[Question 6: 17](#_Toc181914708)

[17](#_Toc181914709)

[Question 7: 18](#_Toc181914710)

[18](#_Toc181914711)

[Question 8: 19](#_Toc181914712)

[Question 9: 20](#_Toc181914713)

[Question 10: 22](#_Toc181914714)

[SQL CODE: 26](#_Toc181914715)

[Reference List 62](#_Toc181914716)

# Question 1:A screenshot of a computer Description automatically generated

# Question 2:

# A screenshot of a computer Description automatically generated

(IIE, 2024).

# Question 3:

# A screenshot of a computer Description automatically generated

(IIE, 2024).

# Question 4:

# A screenshot of a computer Description automatically generated

(IIE, 2024).

# Question 5:A screenshot of a computer Description automatically generatedA screenshot of a computer Description automatically generated

(IIE, 2024).

# Question 6:

# A screenshot of a computer program Description automatically generated

(IIE, 2024).

# Question 7:

# A screenshot of a computer Description automatically generated

(IIE, 2024).

# Question 8:

A screenshot of a computer

Description automatically generated

# Question 9: A screenshot of a computer Description automatically generatedA screenshot of a computer Description automatically generated

(IIE, 2024).

# Question 10:

**Technical Report on Maintaining BikesRUs' Availability, Integrity, and Confidentiality.**

Ensuring the security, accuracy, and consistency of data is crucial when establishing a database system for BikesRUs. A framework for accomplishing this is provided by the ideas of Confidentiality, Integrity, and Availability (CIA). A thorough explanation of how to apply these ideas at BikesRUs, complete with relevant resources, methods, and examples, is provided below (dev, 2021).

**1. Confidentiality**

Confidentiality guards against unwanted access and guarantees that only authorized personnel can access sensitive data. Donor information, volunteer information, and transaction records are examples of sensitive data for BikesRUs. Here are some ways to protect confidentiality (rubrik, 2024).

Role-based access and access control restricts data access according to a user's role, access control mechanisms ought to be put in place. Sales volunteers, for instance, might only require access to data pertaining to customers, whereas donation handlers ought to have access to donor and donation records. By putting role-based access control (RBAC) into practice, you can make sure that each user group can access only the information required for their role. This can be set up in SQL by assigning particular table access according to user roles using GRANT statements (rubrik, 2024).

**Code Example:**

**GRANT SELECT ON Donations TO Sales\_Volunteer;**

**GRANT SELECT, INSERT ON Donations TO Donation\_Manager;**

With data encryption maintaining confidentiality requires encrypting sensitive data both in transit (such as over a network) and at rest (such as in a database). Databases such as Oracle and SQL Server support Transparent Data Encryption (TDE), which encrypts stored data without requiring modifications to the application. For example, the Donors table's sensitive fields, such as addresses or phone numbers, can be encrypted to lower the possibility of unintentional access to the data in the database.

Views for data concealing are for users who need restricted data visibility, views can assist in limiting and masking data. To ensure confidentiality for the Donor table data, a view could by showing donor names and bike types while concealing contact information.

**Code Example:**

CREATE VIEW vwLimitedDonorInfo AS

SELECT DONOR\_ID, DONOR\_FNAME, DONOR\_LNAME, BIKE\_TYPE

FROM Donors

JOIN Bikes ON Donors.DONOR\_ID = Bikes.DONOR\_ID;

**2.  Integrity**

Throughout the database lifecycle, data integrity guarantees that the information is reliable, precise, and trustworthy. Ensuring the accuracy of bike data, volunteer information, and donation records is crucial for BikesRUs (dev, 2021). The techniques to preserve data integrity are listed below.

Foreign keys, unique constraints, and primary keys are examples of constraints. To ensure data consistency, use constraints. While foreign keys preserve relationships between tables, primary keys guarantee that every entry in a table is unique. Setting foreign keys stops donation entries for non-existent donors or bikes because the Donations table refers to the Donors and Bikes tables.**Code Example:**

ALTER TABLE Donations

ADD CONSTRAINT FK\_DonorID FOREIGN KEY (DONOR\_ID) REFERENCES Donors(DONOR\_ID);

Data validation Triggers have the ability to enforce intricate integrity regulations. A trigger, for example, can guarantee that bike values in the Donations table are greater than zero, thereby avoiding inaccurate data entry. Since BikesRUs must keep a comprehensive record of all donations, another trigger might stop donation records from being deleted.

**Code Example:**

CREATE TRIGGER PreventNegativeBikeValue

BEFORE INSERT OR UPDATE ON Donations

FOR EACH ROW

WHEN (NEW.VALUE <= 0)

BEGIN

RAISE\_APPLICATION\_ERROR(-20001, 'Bike value must be positive.');

END;

Frequent data Inspection and verifying: Discrepancies or anomalies can be found with the aid of periodic data audits. Inconsistencies can be found and fixed, for instance, by routinely checking the Donations and Bikes tables. By spotting duplicate records, NULL values in required fields, or inconsistent relationships, SQL scripts or automated tools can help with auditing (dev, 2021).

**3. Accessibility**

Availability guarantees that authorized users can access the database at any time. A strong availability strategy is essential because BikesRUs depends on its database to track inventory, manage donations, and record transactions (dev, 2021).

Regular backups are your initial line of protection for data availability, and backup and recovery can help. Periodically, you should make full backups and incremental backups of any changes you make in between. These backups allow for quick restoration in the event of data loss. To guarantee data recovery functionality, computerized backups should be intended and backup restoration tested on a regular basis (rubrik, 2024).

Consider a high-performance setup that includes clustering and database replication for continuous operation. In order to provide an alternative mechanism in the event that the primary server fails, replication entails copying data in real time to a different server. To improve availability, like SQL Server continually on the accessibility teams can spread data across several nodes (rubrik, 2024).

planning for disaster recovery, to specify how to resume operations in the event of a catastrophic event, such as a server crash or a natural disaster, BikesRUs should create a disaster recovery plan. A geographically remote backup location in which data is frequently replicated may be part of this plan.

The optimization of performance and database maintenance affects availability because sluggish query times can cause access issues. The server load can be decreased and query speed increased by using strategies like indexing fields that are frequently queried, such as bike IDs and donation dates (rubrik, 2024). Database optimization is maintained through routine maintenance tasks like updating database statistics and rebuilding indexes (dev, 2021).

**Tools and Platform Examples:**

SQL Server or Data Guard by Oracle Teams that are always available as these high-availability tools are intended to recover and replicate data in the event of a server failure. They enable the main and backup servers to synchronize in almost real-time.

Cloud-based database services like AWS RDS and Azure SQL Database come with built-in tools for high availability, scalability, and backup. Since both platforms offer computerized backup, the use of encryption, and adherence features, BikesRUs may want to use them if they intend to migrate to the cloud (rubrik, 2024).

# SQL CODE:

-- ST10393280 Daniel Luke James

-- Question 1

-- New database schema

CREATE DATABASE BikesRUs;

USE BikesRUs;

-- Table Creation: Volunteers

CREATE TABLE Volunteers (

VOL\_ID VARCHAR(10) PRIMARY KEY,

VOL\_FNAME VARCHAR(50),

VOL\_SNAME VARCHAR(50),

CONTACT VARCHAR(15),

ADDRESS VARCHAR(100),

EMAIL VARCHAR(50)

);

-- Table Creation: Donors

CREATE TABLE Donors (

DONOR\_ID VARCHAR(10) PRIMARY KEY,

DONOR\_FNAME VARCHAR(50),

DONOR\_LNAME VARCHAR(50),

CONTACT\_NO VARCHAR(15),

EMAIL VARCHAR(50)

);

-- Table Creation: Bikes

CREATE TABLE Bikes (

BIKE\_ID VARCHAR(10) PRIMARY KEY,

DESCRIPTION VARCHAR(100),

BIKE\_TYPE VARCHAR(20),

MANUFACTURER VARCHAR(50)

);

-- Table Creation: Donations

CREATE TABLE Donations (

DONATION\_ID INT PRIMARY KEY,

DONOR\_ID VARCHAR(10),

BIKE\_ID VARCHAR(10),

VALUE DECIMAL(10,2),

VOLUNTEER\_ID VARCHAR(10),

DONATION\_DATE DATE,

FOREIGN KEY (DONOR\_ID) REFERENCES Donors(DONOR\_ID),

FOREIGN KEY (BIKE\_ID) REFERENCES Bikes(BIKE\_ID),

FOREIGN KEY (VOLUNTEER\_ID) REFERENCES Volunteers(VOL\_ID)

);

-- Table Creation: Billing

CREATE TABLE Billing (

BILL\_ID INT PRIMARY KEY,

CUSTOMER\_ID INT,

STAFF\_ID VARCHAR(10),

BILL\_DATE DATE

);

-- Table Creation: Customers

CREATE TABLE Customers (

CUSTOMER\_ID INT PRIMARY KEY,

FIRST\_NAME VARCHAR(50),

SURNAME VARCHAR(50),

ADDRESS VARCHAR(100),

PHONE\_NUM VARCHAR(15),

EMAIL VARCHAR(50)

);

-- Table Creation: Delivery Items

CREATE TABLE Delivery\_Items (

DELIVERY\_ITEM\_ID VARCHAR(10) PRIMARY KEY,

DESCRIPTION VARCHAR(100),

STAFF\_ID VARCHAR(10)

);

-- Table Creation: Drivers

CREATE TABLE Drivers (

DRIVER\_ID VARCHAR(10) PRIMARY KEY,

FIRST\_NAME VARCHAR(50),

SURNAME VARCHAR(50),

DRIVER\_CODE VARCHAR(10),

PHONE\_NUM VARCHAR(15),

ADDRESS VARCHAR(100)

);

-- Table Creation: Driver Deliveries

CREATE TABLE Driver\_Deliveries (

DRIVER\_DELIVERY\_ID VARCHAR(10) PRIMARY KEY,

VIN\_NUMBER VARCHAR(20),

DRIVER\_ID VARCHAR(10),

DELIVERY\_ITEM\_ID VARCHAR(10),

FOREIGN KEY (DRIVER\_ID) REFERENCES Drivers(DRIVER\_ID),

FOREIGN KEY (DELIVERY\_ITEM\_ID) REFERENCES Delivery\_Items(DELIVERY\_ITEM\_ID)

);

-- Table Creation: Staff

CREATE TABLE Staff (

STAFF\_ID VARCHAR(10) PRIMARY KEY,

FIRST\_NAME VARCHAR(50),

SURNAME VARCHAR(50),

POSITION VARCHAR(50),

PHONE\_NUM VARCHAR(15),

ADDRESS VARCHAR(100),

EMAIL VARCHAR(50)

);

-- Table Creation: Vehicles

CREATE TABLE Vehicles (

VIN\_NUMBER VARCHAR(20) PRIMARY KEY,

VEHICLE\_TYPE VARCHAR(50),

MILEAGE INT,

COLOUR VARCHAR(20),

MANUFACTURER VARCHAR(50)

);

-- Insert Data for Volunteers

INSERT INTO Volunteers (VOL\_ID, VOL\_FNAME, VOL\_SNAME, CONTACT, ADDRESS, EMAIL) VALUES

('vol101', 'Kenny', 'Temba', '0677277521', '10 Sands Road', 'kennyt@bikerus.com');

INSERT INTO Volunteers (VOL\_ID, VOL\_FNAME, VOL\_SNAME, CONTACT, ADDRESS, EMAIL) VALUES

('vol102', 'Mamelodi', 'Marks', '0737377522', '20 Langes Street', 'mamelodim@bikerus.com');

INSERT INTO Volunteers (VOL\_ID, VOL\_FNAME, VOL\_SNAME, CONTACT, ADDRESS, EMAIL) VALUES

('vol103', 'Ada', 'Andrews', '0817117523', '31 Williams Street', 'adanyaa@bikerus.com');

INSERT INTO Volunteers (VOL\_ID, VOL\_FNAME, VOL\_SNAME, CONTACT, ADDRESS, EMAIL) VALUES

('vol104', 'Evans', 'Tambala', '0697215244', '1 Free Road', 'evanst@bikerus.com');

INSERT INTO Volunteers (VOL\_ID, VOL\_FNAME, VOL\_SNAME, CONTACT, ADDRESS, EMAIL) VALUES

('vol105', 'Xolani', 'Samson', '0727122255', '12 Main Road', 'xolanis@bikerus.com');

-- Insert Data for Donors

INSERT INTO Donors (DONOR\_ID, DONOR\_FNAME, DONOR\_LNAME, CONTACT\_NO, EMAIL) VALUES

('DID11', 'Jeff', 'Wanya', '0827172250', 'wanyajeff@ymail.com');

INSERT INTO Donors (DONOR\_ID, DONOR\_FNAME, DONOR\_LNAME, CONTACT\_NO, EMAIL) VALUES

('DID12', 'Sthembeni', 'Pisho', '0837865670', 'sthepisho@ymail.com');

INSERT INTO Donors (DONOR\_ID, DONOR\_FNAME, DONOR\_LNAME, CONTACT\_NO, EMAIL) VALUES

('DID13', 'James', 'Temba', '0878978650', 'jimmy@ymail.com');

INSERT INTO Donors (DONOR\_ID, DONOR\_FNAME, DONOR\_LNAME, CONTACT\_NO, EMAIL) VALUES

('DID14', 'Luramo', 'Misi', '0826575650', 'luramom@ymail.com');

INSERT INTO Donors (DONOR\_ID, DONOR\_FNAME, DONOR\_LNAME, CONTACT\_NO, EMAIL) VALUES

('DID15', 'Abraham', 'Xolani', '0797656430', 'axolani@ymail.com');

INSERT INTO Donors (DONOR\_ID, DONOR\_FNAME, DONOR\_LNAME, CONTACT\_NO, EMAIL) VALUES

('DID16', 'Rumi', 'Jones', '0668899221', 'rjones@true.com');

INSERT INTO Donors (DONOR\_ID, DONOR\_FNAME, DONOR\_LNAME, CONTACT\_NO, EMAIL) VALUES

('DID17', 'Xolani', 'Redo', '0614553389', 'xredo@ymail.com');

INSERT INTO Donors (DONOR\_ID, DONOR\_FNAME, DONOR\_LNAME, CONTACT\_NO, EMAIL) VALUES

('DID18', 'Tenny', 'Stars', '0824228870', 'tenstars@true.com');

INSERT INTO Donors (DONOR\_ID, DONOR\_FNAME, DONOR\_LNAME, CONTACT\_NO, EMAIL) VALUES

('DID19', 'Tiny', 'Rambo', '0715554333', 'trambo@ymail.com');

INSERT INTO Donors (DONOR\_ID, DONOR\_FNAME, DONOR\_LNAME, CONTACT\_NO, EMAIL) VALUES

('DID20', 'Yannick', 'Leons', '0615554323', 'yleons@true.com');

-- Insert Data for Bikes

INSERT INTO Bikes (BIKE\_ID, DESCRIPTION, BIKE\_TYPE, MANUFACTURER) VALUES

('B001', 'BMX AX1', 'Road Bike', 'BMX');

INSERT INTO Bikes (BIKE\_ID, DESCRIPTION, BIKE\_TYPE, MANUFACTURER) VALUES

('B002', 'Giant Domain 1', 'Road Bike', 'Giant');

INSERT INTO Bikes (BIKE\_ID, DESCRIPTION, BIKE\_TYPE, MANUFACTURER) VALUES

('B003', 'Ascent 26In', 'Mountain Bike', 'Raleigh');

INSERT INTO Bikes (BIKE\_ID, DESCRIPTION, BIKE\_TYPE, MANUFACTURER) VALUES

('B004', 'Canyon 6X', 'Kids Bike', 'Canyon');

INSERT INTO Bikes (BIKE\_ID, DESCRIPTION, BIKE\_TYPE, MANUFACTURER) VALUES

('B005', 'Marvel', 'Gravel Road Bike', 'BMX');

INSERT INTO Bikes (BIKE\_ID, DESCRIPTION, BIKE\_TYPE, MANUFACTURER) VALUES

('B006', 'Mountain 21 Speed', 'Mountain Bike', 'BMX');

INSERT INTO Bikes (BIKE\_ID, DESCRIPTION, BIKE\_TYPE, MANUFACTURER) VALUES

('B007', 'Canyon Roadster', 'Road Bike', 'Canyon');

INSERT INTO Bikes (BIKE\_ID, DESCRIPTION, BIKE\_TYPE, MANUFACTURER) VALUES

('B008', 'Legion 101', 'Hybrid Bike', 'BMX');

INSERT INTO Bikes (BIKE\_ID, DESCRIPTION, BIKE\_TYPE, MANUFACTURER) VALUES

('B009', 'Madonna 9', 'Road Bike', 'Trek');

INSERT INTO Bikes (BIKE\_ID, DESCRIPTION, BIKE\_TYPE, MANUFACTURER) VALUES

('B010', 'Comp 2022', 'Mountain Bike', 'Trek');

INSERT INTO Bikes (BIKE\_ID, DESCRIPTION, BIKE\_TYPE, MANUFACTURER) VALUES

('B011', 'BMX AX15', 'Road Bike', 'BMX');

-- Insert Data for Donations

INSERT INTO Donations (DONATION\_ID, DONOR\_ID, BIKE\_ID, VALUE, VOLUNTEER\_ID, DONATION\_DATE) VALUES

(1, 'DID11', 'B001', 1500, 'vol101', TO\_DATE('01-05-23', 'DD-MM-YY'));

INSERT INTO Donations (DONATION\_ID, DONOR\_ID, BIKE\_ID, VALUE, VOLUNTEER\_ID, DONATION\_DATE) VALUES

(2, 'DID12', 'B002', 2500, 'vol101', TO\_DATE('03-05-23', 'DD-MM-YY'));

INSERT INTO Donations (DONATION\_ID, DONOR\_ID, BIKE\_ID, VALUE, VOLUNTEER\_ID, DONATION\_DATE) VALUES

(3, 'DID13', 'B003', 1000, 'vol103', TO\_DATE('03-05-23', 'DD-MM-YY'));

INSERT INTO Donations (DONATION\_ID, DONOR\_ID, BIKE\_ID, VALUE, VOLUNTEER\_ID, DONATION\_DATE) VALUES

(4, 'DID14', 'B004', 1750, 'vol105', TO\_DATE('05-05-23', 'DD-MM-YY'));

INSERT INTO Donations (DONATION\_ID, DONOR\_ID, BIKE\_ID, VALUE, VOLUNTEER\_ID, DONATION\_DATE) VALUES

(5, 'DID15', 'B006', 2000, 'vol101', TO\_DATE('07-05-23', 'DD-MM-YY'));

INSERT INTO Donations (DONATION\_ID, DONOR\_ID, BIKE\_ID, VALUE, VOLUNTEER\_ID, DONATION\_DATE) VALUES

(6, 'DID16', 'B007', 1800, 'vol105', TO\_DATE('09-05-23', 'DD-MM-YY'));

INSERT INTO Donations (DONATION\_ID, DONOR\_ID, BIKE\_ID, VALUE, VOLUNTEER\_ID, DONATION\_DATE) VALUES

(7, 'DID17', 'B008', 1500, 'vol101', TO\_DATE('15-05-23', 'DD-MM-YY'));

INSERT INTO Donations (DONATION\_ID, DONOR\_ID, BIKE\_ID, VALUE, VOLUNTEER\_ID, DONATION\_DATE) VALUES

(8, 'DID18', 'B009', 1500, 'vol103', TO\_DATE('19-05-23', 'DD-MM-YY'));

INSERT INTO Donations (DONATION\_ID, DONOR\_ID, BIKE\_ID, VALUE, VOLUNTEER\_ID, DONATION\_DATE) VALUES

(9, 'DID12', 'B010', 2500, 'vol103', TO\_DATE('25-05-23', 'DD-MM-YY'));

INSERT INTO Donations (DONATION\_ID, DONOR\_ID, BIKE\_ID, VALUE, VOLUNTEER\_ID, DONATION\_DATE) VALUES

(10, 'DID20', 'B005', 3500, 'vol105', TO\_DATE('05-05-23', 'DD-MM-YY'));

INSERT INTO Donations (DONATION\_ID, DONOR\_ID, BIKE\_ID, VALUE, VOLUNTEER\_ID, DONATION\_DATE) VALUES

(11, 'DID19', 'B011', 2500, 'vol103', TO\_DATE('30-05-23', 'DD-MM-YY'));

-- Insert Data for Billing

INSERT INTO Billing (BILL\_ID, CUSTOMER\_ID, STAFF\_ID, BILL\_DATE) VALUES

(800, 11011, '51011', TO\_DATE('06-09-22', 'DD-MM-YY'));

INSERT INTO Billing (BILL\_ID, CUSTOMER\_ID, STAFF\_ID, BILL\_DATE) VALUES

(801, 11012, '51013', TO\_DATE('07-09-22', 'DD-MM-YY'));

INSERT INTO Billing (BILL\_ID, CUSTOMER\_ID, STAFF\_ID, BILL\_DATE) VALUES

(802, 11014, '51015', TO\_DATE('10-11-22', 'DD-MM-YY'));

INSERT INTO Billing (BILL\_ID, CUSTOMER\_ID, STAFF\_ID, BILL\_DATE) VALUES

(803, 11015, '51012', TO\_DATE('09-12-22', 'DD-MM-YY'));

INSERT INTO Billing (BILL\_ID, CUSTOMER\_ID, STAFF\_ID, BILL\_DATE) VALUES

(804, 11013, '51014', TO\_DATE('09-12-22', 'DD-MM-YY'));

INSERT INTO Billing (BILL\_ID, CUSTOMER\_ID, STAFF\_ID, BILL\_DATE) VALUES

(805, 11111, '51011', TO\_DATE('06-09-22', 'DD-MM-YY'));

INSERT INTO Billing (BILL\_ID, CUSTOMER\_ID, STAFF\_ID, BILL\_DATE) VALUES

(806, 11012, '51013', TO\_DATE('07-09-22', 'DD-MM-YY'));

INSERT INTO Billing (BILL\_ID, CUSTOMER\_ID, STAFF\_ID, BILL\_DATE) VALUES

(807, 11014, '51015', TO\_DATE('10-11-22', 'DD-MM-YY'));

INSERT INTO Billing (BILL\_ID, CUSTOMER\_ID, STAFF\_ID, BILL\_DATE) VALUES

(808, 11015, '51012', TO\_DATE('09-12-22', 'DD-MM-YY'));

INSERT INTO Billing (BILL\_ID, CUSTOMER\_ID, STAFF\_ID, BILL\_DATE) VALUES

(809, 11113, '51018', TO\_DATE('09-12-22', 'DD-MM-YY'));

INSERT INTO Billing (BILL\_ID, CUSTOMER\_ID, STAFF\_ID, BILL\_DATE) VALUES

(810, 11011, '51011', TO\_DATE('06-09-22', 'DD-MM-YY'));

INSERT INTO Billing (BILL\_ID, CUSTOMER\_ID, STAFF\_ID, BILL\_DATE) VALUES

(811, 11012, '51013', TO\_DATE('07-09-22', 'DD-MM-YY'));

INSERT INTO Billing (BILL\_ID, CUSTOMER\_ID, STAFF\_ID, BILL\_DATE) VALUES

(812, 11014, '51016', TO\_DATE('10-11-22', 'DD-MM-YY'));

INSERT INTO Billing (BILL\_ID, CUSTOMER\_ID, STAFF\_ID, BILL\_DATE) VALUES

(813, 11117, '51012', TO\_DATE('09-12-22', 'DD-MM-YY'));

INSERT INTO Billing (BILL\_ID, CUSTOMER\_ID, STAFF\_ID, BILL\_DATE) VALUES

(814, 11013, '51014', TO\_DATE('09-12-22', 'DD-MM-YY'));

INSERT INTO Billing (BILL\_ID, CUSTOMER\_ID, STAFF\_ID, BILL\_DATE) VALUES

(815, 11012, '51111', TO\_DATE('06-09-22', 'DD-MM-YY'));

INSERT INTO Billing (BILL\_ID, CUSTOMER\_ID, STAFF\_ID, BILL\_DATE) VALUES

(816, 11012, '51019', TO\_DATE('07-09-22', 'DD-MM-YY'));

INSERT INTO Billing (BILL\_ID, CUSTOMER\_ID, STAFF\_ID, BILL\_DATE) VALUES

(817, 11014, '51015', TO\_DATE('10-11-22', 'DD-MM-YY'));

INSERT INTO Billing (BILL\_ID, CUSTOMER\_ID, STAFF\_ID, BILL\_DATE) VALUES

(818, 11112, '51012', TO\_DATE('09-12-22', 'DD-MM-YY'));

INSERT INTO Billing (BILL\_ID, CUSTOMER\_ID, STAFF\_ID, BILL\_DATE) VALUES

(819, 11013, '51014', TO\_DATE('09-12-22', 'DD-MM-YY'));

INSERT INTO Billing (BILL\_ID, CUSTOMER\_ID, STAFF\_ID, BILL\_DATE) VALUES

(820, 11116, '51019', TO\_DATE('09-12-22', 'DD-MM-YY'));

-- Insert Data for Customers

INSERT INTO Customers (CUSTOMER\_ID, FIRST\_NAME, SURNAME, ADDRESS, PHONE\_NUM, EMAIL) VALUES

(11011, 'Bob', 'Smith', '18 Water rd', '0877277521', 'bobs@isat.com');

INSERT INTO Customers (CUSTOMER\_ID, FIRST\_NAME, SURNAME, ADDRESS, PHONE\_NUM, EMAIL) VALUES

(11012, 'Sam', 'Hendricks', '22 Water rd', '0863257857', 'shen@mcom.co.za');

INSERT INTO Customers (CUSTOMER\_ID, FIRST\_NAME, SURNAME, ADDRESS, PHONE\_NUM, EMAIL) VALUES

(11013, 'Larry', 'Clark', '101 Summer lane', '0834567891', 'larc@mcom.co.za');

INSERT INTO Customers (CUSTOMER\_ID, FIRST\_NAME, SURNAME, ADDRESS, PHONE\_NUM, EMAIL) VALUES

(11014, 'Jeff', 'Jones', '55 Mountain way', '0612547895', 'jj@isat.co.za');

INSERT INTO Customers (CUSTOMER\_ID, FIRST\_NAME, SURNAME, ADDRESS, PHONE\_NUM, EMAIL) VALUES

(11015, 'Andre', 'Kerk', '5 Main rd', '0827238521', 'akerk@mcal.co.za');

INSERT INTO Customers (CUSTOMER\_ID, FIRST\_NAME, SURNAME, ADDRESS, PHONE\_NUM, EMAIL) VALUES

(11016, 'Wayne', 'Smith', '13 Water rd', '0877277522', 'ws@isat.com');

INSERT INTO Customers (CUSTOMER\_ID, FIRST\_NAME, SURNAME, ADDRESS, PHONE\_NUM, EMAIL) VALUES

(11017, 'John', 'Hendricks', '29 Water rd', '0863257851', 'jhen@mcom.co.za');

INSERT INTO Customers (CUSTOMER\_ID, FIRST\_NAME, SURNAME, ADDRESS, PHONE\_NUM, EMAIL) VALUES

(11018, 'Sally', 'Clark', '111 Summer lane', '0834567892', 'sallyc@mcom.co.za');

INSERT INTO Customers (CUSTOMER\_ID, FIRST\_NAME, SURNAME, ADDRESS, PHONE\_NUM, EMAIL) VALUES

(11019, 'Bridget', 'Bitterhour', '125 Mountain way', '0612547896', 'bb@isat.co.za');

INSERT INTO Customers (CUSTOMER\_ID, FIRST\_NAME, SURNAME, ADDRESS, PHONE\_NUM, EMAIL) VALUES

(11111, 'Nicole', 'Kerk', '175 Main rd', '0827238529', 'nk@mcal.co.za');

INSERT INTO Customers (CUSTOMER\_ID, FIRST\_NAME, SURNAME, ADDRESS, PHONE\_NUM, EMAIL) VALUES

(11112, 'Catherine', 'Smith', '5 Walter rd', '0877277532', 'cath@isat.co.za');

INSERT INTO Customers (CUSTOMER\_ID, FIRST\_NAME, SURNAME, ADDRESS, PHONE\_NUM, EMAIL) VALUES

(11113, 'Mel', 'Hendricks', '5 Water rd', '0863257852', 'melh@mcom.co.za');

INSERT INTO Customers (CUSTOMER\_ID, FIRST\_NAME, SURNAME, ADDRESS, PHONE\_NUM, EMAIL) VALUES

(11114, 'Lucy', 'Du Plessis', '221 Summer lane', '0834567892', 'ldup@mcom.co.za');

INSERT INTO Customers (CUSTOMER\_ID, FIRST\_NAME, SURNAME, ADDRESS, PHONE\_NUM, EMAIL) VALUES

(11116, 'Josh', 'Mavrick', '155 Mountain way', '0612547897', 'joshm@isat.co.za');

INSERT INTO Customers (CUSTOMER\_ID, FIRST\_NAME, SURNAME, ADDRESS, PHONE\_NUM, EMAIL) VALUES

(11117, 'Stuart', 'Jones', '35 Main rd', '0827238521', 'sjohnes@mcal.co.za');

-- Insert Data for Delivery\_Items

INSERT INTO Delivery\_Items (DELIVERY\_ITEM\_ID, DESCRIPTION, STAFF\_ID) VALUES

('71011', 'House relocation', '51011');

INSERT INTO Delivery\_Items (DELIVERY\_ITEM\_ID, DESCRIPTION, STAFF\_ID) VALUES

('71012', 'Delivery of specialized consignments', '51017');

INSERT INTO Delivery\_Items (DELIVERY\_ITEM\_ID, DESCRIPTION, STAFF\_ID) VALUES

('71013', 'Delivery of specialized consignments', '51015');

INSERT INTO Delivery\_Items (DELIVERY\_ITEM\_ID, DESCRIPTION, STAFF\_ID) VALUES

('71014', 'Office relocation', '51012');

INSERT INTO Delivery\_Items (DELIVERY\_ITEM\_ID, DESCRIPTION, STAFF\_ID) VALUES

('71015', 'Delivery of specialized consignments', '51014');

-- Insert Data for Drivers

INSERT INTO Drivers (DRIVER\_ID, FIRST\_NAME, SURNAME, DRIVER\_CODE, PHONE\_NUM, ADDRESS) VALUES

('81011', 'Buthelezi', 'Marshall', 'C1', '725698547', '18 Leopard Creek');

INSERT INTO Drivers (DRIVER\_ID, FIRST\_NAME, SURNAME, DRIVER\_CODE, PHONE\_NUM, ADDRESS) VALUES

('81012', 'Tina', 'Mtati', 'C', '636984178', '12 Cape Rd');

INSERT INTO Drivers (DRIVER\_ID, FIRST\_NAME, SURNAME, DRIVER\_CODE, PHONE\_NUM, ADDRESS) VALUES

('81013', 'Jono', 'Mvuyisi', 'EC1', '725648965', '15 Circle Lane');

INSERT INTO Drivers (DRIVER\_ID, FIRST\_NAME, SURNAME, DRIVER\_CODE, PHONE\_NUM, ADDRESS) VALUES

('81014', 'Richard', 'Smith', 'C1', '623116598', '18 Beach Rd');

INSERT INTO Drivers (DRIVER\_ID, FIRST\_NAME, SURNAME, DRIVER\_CODE, PHONE\_NUM, ADDRESS) VALUES

('81015', 'Brett', 'Smith', 'EB', '883521457', '55 Summer Lane');

-- Insert Data for Driver\_Deliveries

INSERT INTO Driver\_Deliveries (DRIVER\_DELIVERY\_ID, VIN\_NUMBER, DRIVER\_ID, DELIVERY\_ITEM\_ID) VALUES

('91011', '1ZA55858541', '81011', '71011');

INSERT INTO Driver\_Deliveries (DRIVER\_DELIVERY\_ID, VIN\_NUMBER, DRIVER\_ID, DELIVERY\_ITEM\_ID) VALUES

('91012', '1ZA35858543', '81012', '71013');

INSERT INTO Driver\_Deliveries (DRIVER\_DELIVERY\_ID, VIN\_NUMBER, DRIVER\_ID, DELIVERY\_ITEM\_ID) VALUES

('91013', '1ZA17851545', '81011', '71015');

INSERT INTO Driver\_Deliveries (DRIVER\_DELIVERY\_ID, VIN\_NUMBER, DRIVER\_ID, DELIVERY\_ITEM\_ID) VALUES

('91014', '1ZA35868540', '81013', '71015');

INSERT INTO Driver\_Deliveries (DRIVER\_DELIVERY\_ID, VIN\_NUMBER, DRIVER\_ID, DELIVERY\_ITEM\_ID) VALUES

('91015', '1ZA15851545', '81014', '71012');

-- Insert Data for Staff

INSERT INTO Staff (STAFF\_ID, FIRST\_NAME, SURNAME, POSITION, PHONE\_NUM, ADDRESS, EMAIL) VALUES

('51011', 'Sally', 'Du Toit', 'Logistics', '0825698547', '18 Main rd', 'sdut@isat.com');

INSERT INTO Staff (STAFF\_ID, FIRST\_NAME, SURNAME, POSITION, PHONE\_NUM, ADDRESS, EMAIL) VALUES

('51012', 'Mark', 'Wright', 'CRM', '0836984178', '12 Cape way', 'mwright@isat.com');

INSERT INTO Staff (STAFF\_ID, FIRST\_NAME, SURNAME, POSITION, PHONE\_NUM, ADDRESS, EMAIL) VALUES

('51013', 'Harry', 'Sheen', 'Logistics', '0725648965', '15 Water st', 'hsheen@isat.com');

INSERT INTO Staff (STAFF\_ID, FIRST\_NAME, SURNAME, POSITION, PHONE\_NUM, ADDRESS, EMAIL) VALUES

('51014', 'Jabu', 'Xolani', 'Logistics', '0823116598', '18 White lane', 'jxo@isat.com');

INSERT INTO Staff (STAFF\_ID, FIRST\_NAME, SURNAME, POSITION, PHONE\_NUM, ADDRESS, EMAIL) VALUES

('51015', 'Roberto', 'Henry', 'Packaging', '0783521451', '55 Cape st', 'rhenry@isat.com');

INSERT INTO Staff (STAFF\_ID, FIRST\_NAME, SURNAME, POSITION, PHONE\_NUM, ADDRESS, EMAIL) VALUES

('51016', 'Pat', 'Durant', 'Logistics', '0825698542', '1 Main rd', 'pd@isat.com');

INSERT INTO Staff (STAFF\_ID, FIRST\_NAME, SURNAME, POSITION, PHONE\_NUM, ADDRESS, EMAIL) VALUES

('51017', 'Steve', 'Maritz', 'CRM', '0836984173', '2 Cape way', 'sm@isat.com');

INSERT INTO Staff (STAFF\_ID, FIRST\_NAME, SURNAME, POSITION, PHONE\_NUM, ADDRESS, EMAIL) VALUES

('51018', 'Maxwell', 'Dube', 'Logistics', '0725648964', '5 Water st', 'max@isat.com');

INSERT INTO Staff (STAFF\_ID, FIRST\_NAME, SURNAME, POSITION, PHONE\_NUM, ADDRESS, EMAIL) VALUES

('51019', 'Shane', 'Mane', 'Logistics', '0823116595', '8 White lane', 'smane@isat.com');

INSERT INTO Staff (STAFF\_ID, FIRST\_NAME, SURNAME, POSITION, PHONE\_NUM, ADDRESS, EMAIL) VALUES

('51111', 'Bob', 'Truth', 'Packaging', '0783521456', '35 Cape st', 'btruth@isat.com');

-- Insert Data for Vehicles

INSERT INTO Vehicles (VIN\_NUMBER, VEHICLE\_TYPE, MILEAGE, COLOUR, MANUFACTURER) VALUES

('1ZA55858541', 'Cutaway Van Chassis', 115352, 'Red', 'MAN');

INSERT INTO Vehicles (VIN\_NUMBER, VEHICLE\_TYPE, MILEAGE, COLOUR, MANUFACTURER) VALUES

('1ZA51858542', 'Flatbed Truck', 315856, 'Blue', 'ISUZU');

INSERT INTO Vehicles (VIN\_NUMBER, VEHICLE\_TYPE, MILEAGE, COLOUR, MANUFACTURER) VALUES

('1ZA35858543', 'Medium Standard Truck', 789587, 'Silver', 'MAN');

INSERT INTO Vehicles (VIN\_NUMBER, VEHICLE\_TYPE, MILEAGE, COLOUR, MANUFACTURER) VALUES

('1ZA15851545', 'Flatbed Truck', 555050, 'White', 'TATA');

INSERT INTO Vehicles (VIN\_NUMBER, VEHICLE\_TYPE, MILEAGE, COLOUR, MANUFACTURER) VALUES

('1ZA35868540', 'Cutaway Van Chassis', 79058, 'White', 'ISUZU');

INSERT INTO Vehicles (VIN\_NUMBER, VEHICLE\_TYPE, MILEAGE, COLOUR, MANUFACTURER) VALUES

('1ZA65858541', 'Cutaway Van Chassis', 215352, 'Red', 'MAN');

INSERT INTO Vehicles (VIN\_NUMBER, VEHICLE\_TYPE, MILEAGE, COLOUR, MANUFACTURER) VALUES

('1ZA61858542', 'Flatbed Truck', 215856, 'Blue', 'ISUZU');

INSERT INTO Vehicles (VIN\_NUMBER, VEHICLE\_TYPE, MILEAGE, COLOUR, MANUFACTURER) VALUES

('1ZA65858543', 'Medium Standard Truck', 889587, 'Silver', 'MERC');

INSERT INTO Vehicles (VIN\_NUMBER, VEHICLE\_TYPE, MILEAGE, COLOUR, MANUFACTURER) VALUES

('1ZA65851545', 'Flatbed Truck', 155050, 'White', 'MAN');

INSERT INTO Vehicles (VIN\_NUMBER, VEHICLE\_TYPE, MILEAGE, COLOUR, MANUFACTURER) VALUES

('1ZA65868540', 'Cutaway Van Chassis', 19058, 'White', 'ISUZU');

INSERT INTO Vehicles (VIN\_NUMBER, VEHICLE\_TYPE, MILEAGE, COLOUR, MANUFACTURER) VALUES

('1ZA75858541', 'Cutaway Van Chassis', 315352, 'Red', 'MAN');

INSERT INTO Vehicles (VIN\_NUMBER, VEHICLE\_TYPE, MILEAGE, COLOUR, MANUFACTURER) VALUES

('1ZA71858542', 'Flatbed Truck', 115856, 'Blue', 'ISUZU');

INSERT INTO Vehicles (VIN\_NUMBER, VEHICLE\_TYPE, MILEAGE, COLOUR, MANUFACTURER) VALUES

('1ZA75858543', 'Medium Standard Truck', 989587, 'Silver', 'MAN');

INSERT INTO Vehicles (VIN\_NUMBER, VEHICLE\_TYPE, MILEAGE, COLOUR, MANUFACTURER) VALUES

('1ZA17851545', 'Flatbed Truck', 755050, 'White', 'TATA');

INSERT INTO Vehicles (VIN\_NUMBER, VEHICLE\_TYPE, MILEAGE, COLOUR, MANUFACTURER) VALUES

('1ZA75868540', 'Cutaway Van Chassis', 29058, 'White', 'ISUZU');

INSERT INTO Vehicles (VIN\_NUMBER, VEHICLE\_TYPE, MILEAGE, COLOUR, MANUFACTURER) VALUES

('1ZA85858541', 'Cutaway Van Chassis', 515352, 'Red', 'MERC');

INSERT INTO Vehicles (VIN\_NUMBER, VEHICLE\_TYPE, MILEAGE, COLOUR, MANUFACTURER) VALUES

('1ZA81858542', 'Flatbed Truck', 715856, 'Blue', 'ISUZU');

INSERT INTO Vehicles (VIN\_NUMBER, VEHICLE\_TYPE, MILEAGE, COLOUR, MANUFACTURER) VALUES

('1ZA85858543', 'Medium Standard Truck', 789587, 'Silver', 'MAN');

INSERT INTO Vehicles (VIN\_NUMBER, VEHICLE\_TYPE, MILEAGE, COLOUR, MANUFACTURER) VALUES

('1ZA85851545', 'Flatbed Truck', 955050, 'White', 'TATA');

INSERT INTO Vehicles (VIN\_NUMBER, VEHICLE\_TYPE, MILEAGE, COLOUR, MANUFACTURER) VALUES

('1ZA85868540', 'Cutaway Van Chassis', 39058, 'White', 'MERC');

-- View data tables

SELECT \* FROM Volunteers;

SELECT \* FROM Donors;

SELECT \* FROM Bikes;

SELECT \* FROM Donations;

SELECT \* FROM Billing;

SELECT \* FROM Customers;

SELECT \* FROM Delivery\_Items;

SELECT \* FROM Drivers;

SELECT \* FROM Driver\_Deliveries;

SELECT \* FROM Staff;

SELECT \* FROM Vehicles;

-- Question 2

SELECT d.DONOR\_ID, b.BIKE\_TYPE, b.DESCRIPTION, dn.VALUE

FROM Donors d

JOIN Donations dn ON d.DONOR\_ID = dn.DONOR\_ID

JOIN Bikes b ON dn.BIKE\_ID = b.BIKE\_ID

WHERE dn.VALUE > 1500;

-- Question 3

-- Set DBMS OUTPUT

SET SERVEROUTPUT ON SIZE 1000000

DECLARE

-- Declare the VAT constant as 15%

v\_vat CONSTANT NUMBER := 0.15;

-- Declare variables to store the bike information

v\_description Bikes.DESCRIPTION%TYPE;

v\_manufacturer Bikes.MANUFACTURER%TYPE;

v\_bike\_type Bikes.BIKE\_TYPE%TYPE;

v\_value Donations.VALUE%TYPE;

v\_vat\_amount NUMBER;

v\_total\_amount NUMBER;

-- Cursor to retrieve details of road bikes only

CURSOR road\_bike\_cursor IS

SELECT b.DESCRIPTION, b.MANUFACTURER, b.BIKE\_TYPE, dn.VALUE

FROM Bikes b

JOIN Donations dn ON b.BIKE\_ID = dn.BIKE\_ID

WHERE b.BIKE\_TYPE = 'Road Bike';

BEGIN

-- Open the cursor and loop through each road bike record

OPEN road\_bike\_cursor;

LOOP

FETCH road\_bike\_cursor INTO v\_description, v\_manufacturer, v\_bike\_type, v\_value;

EXIT WHEN road\_bike\_cursor%NOTFOUND;

-- Calculate VAT and total amount

v\_vat\_amount := v\_value \* v\_vat;

v\_total\_amount := v\_value + v\_vat\_amount;

-- Output the results in the specified format

DBMS\_OUTPUT.PUT\_LINE('BIKE DESCRIPTION: ' || v\_description);

DBMS\_OUTPUT.PUT\_LINE('BIKE MANUFACTURER: ' || v\_manufacturer);

DBMS\_OUTPUT.PUT\_LINE('BIKE TYPE: ' || v\_bike\_type);

DBMS\_OUTPUT.PUT\_LINE('VALUE: R' || TO\_CHAR(v\_value, '999,999.99'));

DBMS\_OUTPUT.PUT\_LINE('VAT: R' || TO\_CHAR(v\_vat\_amount, '999,999.99'));

DBMS\_OUTPUT.PUT\_LINE('TOTAL AMNT: R' || TO\_CHAR(v\_total\_amount, '999,999.99'));

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

END LOOP;

-- Close the cursor

CLOSE road\_bike\_cursor;

END;

-- Question 4

-- Create or replace the view vwBikeRUs

CREATE OR REPLACE VIEW vwBikeRUs AS

SELECT

d.DONOR\_FNAME || ' ' || d.DONOR\_LNAME AS DONOR\_NAME,

d.CONTACT\_NO AS DONOR\_CONTACT,

b.BIKE\_TYPE,

dn.DONATION\_DATE

FROM Donors d

JOIN Donations dn ON d.DONOR\_ID = dn.DONOR\_ID

JOIN Bikes b ON dn.BIKE\_ID = b.BIKE\_ID

WHERE dn.VOLUNTEER\_ID = 'vol105';

-- View to display the results

SELECT \* FROM vwBikeRUs;

/\*

Justification for Using a View:

1) Data Security: Views give the outlet the ability to limit access to private information.

This view improves data security by allowing users to examine only pertinent donor and bike information without requiring access to the full underlying tables.

2) Simplified Querying: Views provide complicated queries a more straightforward user interface.

For instance, users can easily receive combined donor and bike information for a particular volunteer by utilizing `vwBikeRUs`,

which simplifies data retrieval and lowers the possibility of errors, without having to write complicated joins each time.

\*/

-- Question 5

-- Solution 1: Correct Output

CREATE OR REPLACE PROCEDURE spDonorDetails (p\_bike\_id IN VARCHAR2) AS

v\_donor\_name VARCHAR2(100);

v\_contact\_no VARCHAR2(15);

v\_volunteer\_name VARCHAR2(50);

v\_donation\_date DATE;

BEGIN

-- Fetch donor, volunteer, and donation details for the provided bike ID

SELECT d.DONOR\_FNAME || ' ' || d.DONOR\_LNAME, d.CONTACT\_NO, v.VOL\_FNAME, dn.DONATION\_DATE

INTO v\_donor\_name, v\_contact\_no, v\_volunteer\_name, v\_donation\_date

FROM Donors d

JOIN Donations dn ON d.DONOR\_ID = dn.DONOR\_ID

JOIN Volunteers v ON v.VOL\_ID = dn.VOLUNTEER\_ID

WHERE dn.BIKE\_ID = p\_bike\_id;

-- Display the fetched details

DBMS\_OUTPUT.PUT\_LINE('Donor Name: ' || v\_donor\_name);

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

DBMS\_OUTPUT.PUT\_LINE('Volunteer Name: ' || v\_volunteer\_name);

DBMS\_OUTPUT.PUT\_LINE('Donation Date: ' || TO\_CHAR(v\_donation\_date, 'DD-MM-YYYY'));

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('No records found for the provided Bike ID.');

WHEN OTHERS THEN

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

END spDonorDetails;

/

-- Execute Procedure for Bike ID 'B004'

BEGIN

spDonorDetails('B004');

END;

/

-- Solution 2: Customized Output

CREATE OR REPLACE PROCEDURE spDonorDetails\_v2 (p\_bike\_id IN VARCHAR2) AS

-- Declare variables

v\_donor\_name VARCHAR2(100);

v\_contact\_no VARCHAR2(15);

v\_volunteer\_name VARCHAR2(50);

v\_donation\_date DATE;

BEGIN

-- Select statement to retrieve data

SELECT d.DONOR\_FNAME || ' ' || d.DONOR\_LNAME, d.CONTACT\_NO, v.VOL\_FNAME, dn.DONATION\_DATE

INTO v\_donor\_name, v\_contact\_no, v\_volunteer\_name, v\_donation\_date

FROM Donors d

JOIN Donations dn ON d.DONOR\_ID = dn.DONOR\_ID

JOIN Volunteers v ON v.VOL\_ID = dn.VOLUNTEER\_ID

WHERE dn.BIKE\_ID = p\_bike\_id;

-- Output result in a custom format

DBMS\_OUTPUT.PUT\_LINE('### Donor and Volunteer Details ###');

DBMS\_OUTPUT.PUT\_LINE('Donor: ' || v\_donor\_name || ' | Contact: ' || v\_contact\_no);

DBMS\_OUTPUT.PUT\_LINE('Volunteer: ' || v\_volunteer\_name || ' | Date: ' || TO\_CHAR(v\_donation\_date, 'DD-MON-YYYY'));

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('No data found for Bike ID ' || p\_bike\_id);

WHEN OTHERS THEN

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

END spDonorDetails\_v2;

/

-- Execute Procedure for Bike ID 'B004'

BEGIN

spDonorDetails\_v2('B004');

END;

/

-- Question 6

CREATE OR REPLACE FUNCTION fn\_TotalDonations (p\_donor\_id VARCHAR2) RETURN NUMBER IS

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

BEGIN

-- Calculate total donation amount

SELECT SUM(VALUE) INTO v\_total\_donations FROM Donations WHERE DONOR\_ID = p\_donor\_id;

RETURN NVL(v\_total\_donations, 0);

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN 0; -- Return 0 if no donations found

WHEN OTHERS THEN

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

RETURN 0;

END fn\_TotalDonations;

/

-- Execute Function

DECLARE

v\_donor\_id VARCHAR2(10) := 'DID11';

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

BEGIN

v\_total\_donated := fn\_TotalDonations(v\_donor\_id);

DBMS\_OUTPUT.PUT\_LINE('Total Donations by Donor ' || v\_donor\_id || ': R' || v\_total\_donated);

END;

/

-- Question 7

DECLARE

CURSOR bike\_cursor IS

SELECT b.BIKE\_ID, b.BIKE\_TYPE, b.MANUFACTURER, d.VALUE

FROM Bikes b

JOIN Donations d ON b.BIKE\_ID = d.BIKE\_ID;

bike\_status VARCHAR2(10);

BEGIN

FOR bike\_record IN bike\_cursor LOOP

IF bike\_record.VALUE <= 1500 THEN

bike\_status := '\*';

ELSIF bike\_record.VALUE > 1500 AND bike\_record.VALUE <= 3000 THEN

bike\_status := '\*\*';

ELSE

bike\_status := '\*\*\*';

END IF;

DBMS\_OUTPUT.PUT\_LINE('Bike ID: ' || bike\_record.BIKE\_ID);

DBMS\_OUTPUT.PUT\_LINE('Bike Type: ' || bike\_record.BIKE\_TYPE);

DBMS\_OUTPUT.PUT\_LINE('Bike Manufacturer: ' || bike\_record.MANUFACTURER);

DBMS\_OUTPUT.PUT\_LINE('Bike Value: R' || bike\_record.VALUE);

DBMS\_OUTPUT.PUT\_LINE('Status: ' || bike\_status);

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

END LOOP;

END;

/

-- Question 8

SELECT

b.BIKE\_ID,

b.BIKE\_TYPE,

b.MANUFACTURER,

d.VALUE AS BIKE\_VALUE,

CASE

WHEN d.VALUE <= 1500 THEN '\*'

WHEN d.VALUE > 1500 AND d.VALUE <= 3000 THEN '\*\*'

ELSE '\*\*\*'

END AS STATUS

FROM

Bikes b

JOIN

Donations d ON b.BIKE\_ID = d.BIKE\_ID

ORDER BY

b.BIKE\_ID;

-- Question 9

-- Q9.1

-- SYS user does not have access rights to the trigger functions

CREATE OR REPLACE TRIGGER trg\_no\_delete\_donations

BEFORE DELETE ON Donations

FOR EACH ROW

BEGIN

RAISE\_APPLICATION\_ERROR(-20001, 'Deletion from Donations table is not allowed.');

END;

/

-- Test the Trigger

DELETE FROM Donations WHERE DONATION\_ID = 1;

-- Q9.2

-- SYS user does not have access rights to the trigger functions

CREATE OR REPLACE TRIGGER trg\_valid\_donation\_value

BEFORE UPDATE OF VALUE ON Donations

FOR EACH ROW

BEGIN

IF :NEW.VALUE <= 0 THEN

RAISE\_APPLICATION\_ERROR(-20002, 'Donation value must be greater than zero.');

END IF;

END;

/

-- Test the Trigger

UPDATE Donations SET VALUE = -100 WHERE DONATION\_ID = 2;

# Reference List

* IIE. (2024). Advanced Databases Module Manual (First ed.) IIE.
* rubrik, 2024. Database Security: Concepts and Best Practices, 2021.

[Online]. Available at:

<https://www.rubrik.com/insights/database-security>

[Accessed: 2 November 2024].

* dev, 2021. Database Security Best Practices, 11 December 2021.

[Online]. Available at:

<https://dev.to/backendless/database-security-best-practices-4baj>

[Accessed: 4 November 2024].