



ADDB 7311












ADVANCED DATABASES

SUMMATIVE PROJECT

NIVAD RAMDASS

By submitting this assignment, I acknowledge that I have read and understood all the rules as per the terms in the registration contract, in particular the assignment and assessment rules in The IIE Assessment Strategy and Policy (IE009), the intellectual integrity and plagiarism rules in the Intellectual Integrity and Property Rights Policy (IE023), as well as any rules and regulations published in the student portal.

PLAGIARISM STATEMENTS

Statement	Signature	Statement	Signature
I have read the assessment rules provided in this declaration.		I have not shared this assessment with any other student.	
This assessment is my own work.		I have not presented the work of published sources as my own work.	
I have not copied any other student's work in this assessment.		I have correctly cited all my sources of information.	
I have not uploaded the assessment question to any website or App offering assessment assistance.		My referencing is technically correct, consistent, and congruent.	
I have not downloaded my assessment response from a website.		I have acted in an academically honest way in this assessment.	
I have not used any AI tool without reviewing, re-writing, and re-working this information, and referencing any AI tools in my work.			

Question 1

Bikes:

```
-- Create the Bikes table without unique constraint on DESCRIPTION
CREATE TABLE Bikes (
    BIKE_ID VARCHAR2(10) PRIMARY KEY,
    DESCRIPTION VARCHAR2(20) NOT NULL,
    BIKE_TYPE VARCHAR2(20) NOT NULL,
    MANUFACTURER VARCHAR2(20) NOT NULL
);
```

Name	Null?	Type
BIKE_ID	NOT NULL	VARCHAR2(10)
DESCRIPTION	NOT NULL	VARCHAR2(20)
BIKE_TYPE	NOT NULL	VARCHAR2(20)
MANUFACTURER	NOT NULL	VARCHAR2(20)

BIKE_ID	DESCRIPTION	BIKE_TYPE	MANUFACTURER
B001	BMX AX1	Road Bike	BMX
B002	Giant Domain 1	Road Bike	Giant
B003	Ascent 26In	Mountain Bike	Raleigh
B004	Canyon 6X	Kids Bike	Canyon
B005	Marvel	Gravel Road Bike	BMX
B006	Mountain 21 Speed	Mountain Bike	BMX
B007	Canyon Roadster	Road Bike	Canyon
B008	Legion 101	Hybrid Bike	BMX
B009	Madonna 9	Road Bike	Trek
B010	Comp 2022	Mountain Bike	Trek
B011	BMX AX15	Road Bike	BMX

11 rows selected.

Donor:

```
-- Create the Donor table
```

```
CREATE TABLE Donor (  
    DONOR_ID VARCHAR2(10) PRIMARY KEY,  
    DONOR_FNAME VARCHAR2(20) NOT NULL,  
    DONOR_LNAME VARCHAR2(20) NOT NULL,  
    CONTACT_NO VARCHAR2(15),  
    EMAIL VARCHAR2(50)  
);
```

Name	Null?	Type

DONOR_ID	NOT NULL	VARCHAR2(10)
DONOR_FNAME	NOT NULL	VARCHAR2(20)
DONOR_LNAME	NOT NULL	VARCHAR2(20)
CONTACT_NO		VARCHAR2(15)
EMAIL		VARCHAR2(50)

DONOR_ID	DONOR_FNAME	DONOR_LNAME	CONTACT_NO	EMAIL

DID11	Jeff	Wanya	0827172250	wanyajeff@ymail.com
DID12	Sthembeni	Pisho	0837865670	sthepisho@ymail.com
DID13	James	Temba	0878978650	jimmy@ymail.com
DID14	Luramo	Misi	0826575650	luramom@ymail.com
DID15	Abraham	Xolani	0797656430	axolani@ymail.com
DID16	Rumi	Jones	0668899221	rjones@true.com
DID17	Xolani	Redo	0614553389	xredo@ymail.com
DID18	Tenny	Stars	0824228870	tenstars@true.com
DID19	Tiny	Rambo	0715554333	trambo@ymail.com
DID20	Yannick	Leons	0615554323	yleons@true.com

```
10 rows selected.
```

Volunteer:

```
-- Create the Volunteer table
CREATE TABLE Volunteer (
    VOL_ID VARCHAR2(10) PRIMARY KEY,
    VOL_FNAME VARCHAR2(20) NOT NULL,
    VOL_SNAME VARCHAR2(20) NOT NULL,
    CONTACT VARCHAR2(15),
    ADDRESS VARCHAR2(50),
    EMAIL VARCHAR2(50)
);
```

Name	Null?	Type
VOL_ID	NOT NULL	VARCHAR2(10)
VOL_FNAME	NOT NULL	VARCHAR2(20)
VOL_SNAME	NOT NULL	VARCHAR2(20)
CONTACT		VARCHAR2(15)
ADDRESS		VARCHAR2(50)
EMAIL		VARCHAR2(50)

VOL_ID	VOL_FNAME	VOL_SNAME	CONTACT	ADDRESS	EMAIL
vol101	Kenny	Temba	0677277521	10 Sands Road	kennyt@bikerus.com
vol102	Mamelodi	Marks	0737377522	20 Langes Street	mamelodin@bikerus.com
vol103	Ada	Andrews	0817117523	31 Williams Street	adanyaa@bikerus.com
vol104	Evans	Tambala	0697215244	1 Free Road	evanst@bikerus.com
vol105	Xolani	Samson	0727122255	12 Main Road	xolanis@bikerus.com

Donation:

```
-- Create the Donation table with foreign keys
CREATE TABLE Donation (
    DONATION_ID NUMBER PRIMARY KEY,
    DONOR_ID VARCHAR2(10) NOT NULL,
    BIKE_ID VARCHAR2(10) NOT NULL,
    DONATION_VALUE NUMBER NOT NULL,
    VOLUNTEER_ID VARCHAR2(10),
    DONATION_DATE DATE,
    FOREIGN KEY (DONOR_ID) REFERENCES Donor(DONOR_ID),
    FOREIGN KEY (BIKE_ID) REFERENCES Bikes(BIKE_ID),
    FOREIGN KEY (VOLUNTEER_ID) REFERENCES Volunteer(VOL_ID)
);
```

Name	Null?	Type
DONATION_ID	NOT NULL	NUMBER
DONOR_ID	NOT NULL	VARCHAR2(10)
BIKE_ID	NOT NULL	VARCHAR2(10)
DONATION_VALUE	NOT NULL	NUMBER
VOLUNTEER_ID		VARCHAR2(10)
DONATION_DATE		DATE

DONATION_ID	DONOR_ID	BIKE_ID	DONATION_VALUE	VOLUNTEER_ID	DONATION_DATE
1	DID11	B001	1500	vol101	01-MAY-23
2	DID12	B002	2500	vol101	03-MAY-23
3	DID13	B003	1000	vol103	03-MAY-23
4	DID14	B004	1750	vol105	05-MAY-23
5	DID15	B006	2000	vol101	07-MAY-23
6	DID16	B007	1800	vol105	09-MAY-23
7	DID17	B008	1500	vol101	15-MAY-23
8	DID18	B009	1500	vol103	19-MAY-23
9	DID12	B010	2500	vol103	25-MAY-23
10	DID20	B005	3500	vol105	05-MAY-23
11	DID19	B011	2500	vol103	30-MAY-23

11 rows selected.

Question 2

```

123
124 -- Question 2
125 COLUMN "DONATION_VALUE" FORMAT A20
126 SELECT
127     d.DONOR_ID,
128     b.BIKE_TYPE,
129     b.DESCRPTION AS BIKE_DESCRIPTION,
130     'R' || TO_CHAR(dn.DONATION_VALUE, '9,999.00') AS DONATION_VALUE
131 FROM
132     Donation dn
133 JOIN
134     Donor d ON dn.DONOR_ID = d.DONOR_ID
135 JOIN
136     Bikes b ON dn.BIKE_ID = b.BIKE_ID
137 WHERE
138     dn.DONATION_VALUE > 1500;
139

```

11 rows selected.

DONOR_ID	BIKE_TYPE	BIKE_DESCRIPTION	DONATION_VALUE
OID12	Road Bike	Giant Domain 1	R 2,500.00
OID14	Kids Bike	Canyon 6X	R 1,750.00
OID20	Gravel Road Bike	Marvel	R 3,500.00
OID15	Mountain Bike	Mountain 21 Speed	R 2,000.00
OID16	Road Bike	Canyon Roadster	R 1,800.00
OID12	Mountain Bike	Comp 2022	R 2,500.00
OID19	Road Bike	BMX AX15	R 2,500.00

7 rows selected.

Question 3

```

140
141 -- Question 3
142 -- Set line size to accommodate longer rows
143 SET LINESIZE 250
144
145 -- Set column widths to prevent wrapping
146 COLUMN "BIKE DESCRIPTION" FORMAT A19
147 COLUMN "BIKE MANUFACTURER" FORMAT A18
148 COLUMN "BIKE TYPE" FORMAT A15
149 COLUMN "VALUE" FORMAT A12
150 COLUMN "VAT" FORMAT A12
151 COLUMN "TOTAL AMNT" FORMAT A15
152
153 -- Query to display the required information with formatted output
154 WITH VAT_RATE AS (
155     SELECT 0.15 AS RATE FROM dual
156 )
157 SELECT
158     b.DESCRPTION AS "BIKE DESCRIPTION",
159     b.MANUFACTURER AS "BIKE MANUFACTURER",
160     b.BIKE_TYPE AS "BIKE TYPE",
161     'R' || TO_CHAR(dn.DONATION_VALUE, '999G999D00') AS "VALUE",
162     'R' || TO_CHAR(dn.DONATION_VALUE * VAT_RATE.RATE, '999G999D00') AS "VAT",
163     'R' || TO_CHAR(dn.DONATION_VALUE * (1 + VAT_RATE.RATE), '999G999D00') AS "TOTAL AMNT"
164 FROM
165     Donation dn
166 JOIN
167     Bikes b ON dn.BIKE_ID = b.BIKE_ID
168 JOIN
169     VAT_RATE ON 1 = 1
170 WHERE
171     b.BIKE_TYPE = 'Road Bike';
172

```

BIKE DESCRIPTION	BIKE MANUFACTURER	BIKE TYPE	VALUE	VAT	TOTAL AMNT
BMX AX1	BMX	Road Bike	R 1,500.00	R 225.00	R 1,725.00
Giant Domain 1	Giant	Road Bike	R 2,500.00	R 375.00	R 2,875.00
Canyon Roadster	Canyon	Road Bike	R 1,800.00	R 270.00	R 2,070.00
Madonna 9	Trek	Road Bike	R 1,500.00	R 225.00	R 1,725.00
BMX AX15	BMX	Road Bike	R 2,500.00	R 375.00	R 2,875.00

View created.

Question 4

```
74 -- Question 4
75 -- Create the view vwBikeRUs
76 CREATE OR REPLACE VIEW vwBikeRUs AS
77 SELECT
78     d.DONOR_FNAME || ' ' || d.DONOR_LNAME AS DONOR_NAME,
79     d.CONTACT_NO AS DONOR_CONTACT,
80     b.BIKE_TYPE,
81     dn.DONATION_DATE
82 FROM
83     Donation dn
84 JOIN
85     Donor d ON dn.DONOR_ID = d.DONOR_ID
86 JOIN
87     Bikes b ON dn.BIKE_ID = b.BIKE_ID
88 WHERE
89     dn.VOLUNTEER_ID = 'vol105';
90
91 -- Query to run the view
92 COLUMN "DONATION_DATE" FORMAT A15
93 COLUMN "DONOR_NAME" FORMAT A15
94 SELECT * FROM vwBikeRUs;
95
96 -- Benefits of using a View:
97 -- 1. Security and Simplification: Views allow database users to access specific columns without exposing the full underlying table structure.
98 -- For example, by using vwBikeRUs, the outlet can allow access to donor contact information and bike types without exposing all donor details.
99 -- 2. Data Consistency and Ease of Use: Views provide a consistent way to present frequently used or complex queries, simplifying database access for
100 -- The outlet can use vwBikeRUs to quickly retrieve donor information for volunteers like 'vol105' without writing complex joins and filters each
```

View created.

DONOR_NAME	DONOR_CONTACT	BIKE_TYPE	DONATION_DATE
Luramo, Misi	0826575650	Kids Bike	05-MAY-23
Yannick, Leons	0615554323	Gravel Road Bike	05-MAY-23
Rumi, Jones	0668899221	Road Bike	09-MAY-23

Question 5

```
C:\Users\lab_services_student\Documents\adbd_summative\summative.sql - Notepad++ [Administrator]
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
summative.sql
202 -- Question 5
203 SET SERVEROUTPUT ON
204
205 -- Create the procedure spDonorDetails
206 CREATE OR REPLACE PROCEDURE spDonorDetails (p_bike_id IN VARCHAR2) IS
207     v_donor_name    VARCHAR2(40);
208     v_contact_no    VARCHAR2(15);
209     v_volunteer_fname VARCHAR2(20);
210     v_donation_date DATE;
211     v_bike_desc     VARCHAR2(50);
212
213 BEGIN
214     -- Retrieve the required donor, volunteer, and donation information
215     SELECT d.DONOR_FNAME || ' ' || d.DONOR_LNAME,
216            d.CONTACT_NO,
217            v.VOL_FNAME,
218            dn.DONATION_DATE,
219            b.DESCRPTION
220     INTO v_donor_name, v_contact_no, v_volunteer_fname, v_donation_date, v_bike_desc
221     FROM Donation dn
222     JOIN Donor d ON dn.DONOR_ID = d.DONOR_ID
223     JOIN Volunteer v ON dn.VOLUNTEER_ID = v.VOL_ID
224     JOIN Bikes b ON dn.BIKE_ID = b.BIKE_ID
225     WHERE dn.BIKE_ID = p_bike_id;
226
227     -- Display the formatted output
228     DBMS_OUTPUT.PUT_LINE('ATTENTION: ' || v_donor_name ||
229                          ' assisted by: ' || v_volunteer_fname ||
230                          ', donated the ' || v_bike_desc ||
231                          ' on the ' || TO_CHAR(v_donation_date, 'DD/MON/YYYY'));
232
233 EXCEPTION
234     -- Exception handling: In case no data is found for the provided bike ID
235     WHEN NO_DATA_FOUND THEN
236         DBMS_OUTPUT.PUT_LINE('Error: No donation record found for Bike ID ' || p_bike_id);
237     -- Exception handling: In case of any other unexpected errors
238     WHEN OTHERS THEN
239         DBMS_OUTPUT.PUT_LINE('An unexpected error occurred. Error Code: ' || SQLCODE || ' - ' || SQLERRM);
240
241 END spDonorDetails;
242 /
243
244 -- Execute the procedure for Bike ID 'B004'
245 BEGIN
246     spDonorDetails('B004');
247 END;
248 /
249
250
251
Structured Query Language file length: 17,841 lines: 419 Ln: 154 Col: 19 Pos: 8,667 Windows (CR LF) UTF-8 INS
7:41 PM 11/7/2024
```



```
-- Comments on the use of exception handling in this code:
-- 1. NO_DATA_FOUND: This exception handles cases where no donation record is found for the provided Bike ID.
-- It ensures that an informative message is displayed instead of causing an unhandled error.
-- 2. OTHERS: This generic exception handles any unexpected errors that might occur during procedure execution.
-- Displaying the error code and message helps identify the issue for debugging and error tracking.
```

Procedure created.

ATTENTION: Luramo Misi assisted by: Xolani, donated the Canyon 6X on the 05/MAY/2023

Question 6

```
-- Question 6
-- Enable output display in SQL*Plus
SET SERVEROUTPUT ON;

-- Create the function calculateTotalDonations
CREATE OR REPLACE FUNCTION calculateTotalDonations(p_donor_id IN VARCHAR2) RETURN NUMBER IS
    v_total_donations NUMBER := 0;
BEGIN
    -- Calculate total donations for the specified donor ID
    SELECT SUM(dn.DONATION_VALUE)
    INTO v_total_donations
    FROM Donation dn
    WHERE dn.DONOR_ID = p_donor_id;

    -- If the total is NULL, it means the donor has no donations
    IF v_total_donations IS NULL THEN
        DBMS_OUTPUT.PUT_LINE('No donations found for Donor ID: ' || p_donor_id);
        RETURN 0; -- Return 0 if no donations are found
    ELSE
        DBMS_OUTPUT.PUT_LINE('Total donations for Donor ID: ' || p_donor_id || ' is R ' || TO_CHAR(v_total_donations, '9,999.00'));
        RETURN v_total_donations; -- Return the total donation amount
    END IF;

EXCEPTION
    -- Exception handling: Catch unexpected errors
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('An unexpected error occurred. Error Code: ' || SQLCODE || ' - ' || SQLERRM);
        RETURN NULL; -- Return NULL in case of an error
END calculateTotalDonations;
/

-- Execute the function for a specific donor ID
DECLARE
    v_total NUMBER;
BEGIN
    v_total := calculateTotalDonations('DID11'); -- Replace 'DID11' with any valid donor ID to test
    -- Optional: You can display the total if needed
    IF v_total IS NOT NULL THEN
        DBMS_OUTPUT.PUT_LINE('Returned Total: R ' || TO_CHAR(v_total, '9,999.00'));
    END IF;
END;
/
```

Function created.

Total donations for Donor ID: DID11 is R 1,500.00
Returned Total: R 1,500.00

Question 7

```
-- Question 7
-- Enable output display in SQL*Plus
SET SERVEROUTPUT ON;

-- Generate the report for bike statuses
DECLARE
    v_status VARCHAR2(5);
    v_bike_value NUMBER;
BEGIN
    FOR bike_record IN (
        SELECT
            b.BIKE_ID,
            b.BIKE_TYPE,
            b.MANUFACTURER,
            dn.DONATION_VALUE
        FROM
            Bikes b
        JOIN
            Donation dn ON b.BIKE_ID = dn.BIKE_ID
    ) LOOP
        -- Assign status based on donation value
        v_bike_value := bike_record.DONATION_VALUE;

        IF v_bike_value BETWEEN 0 AND 1500 THEN
            v_status := '*'; -- 1-star status
        ELSIF v_bike_value > 1500 AND v_bike_value <= 3000 THEN
            v_status := '**'; -- 2-star status
        ELSIF v_bike_value > 3000 THEN
            v_status := '***'; -- 3-star status
        END IF;

        -- Display the formatted output
        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: ' || TO_CHAR(v_bike_value));
        DBMS_OUTPUT.PUT_LINE('STATUS: ' || v_status);
        DBMS_OUTPUT.PUT_LINE('-----');
    END LOOP;
EXCEPTION
    -- Exception handling: In case of any unexpected errors
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('An unexpected error occurred. Error Code: ' || SQLCODE || ' - ' || SQLERRM);
END;
/
```

PL/SQL procedure successfully completed.

```
BIKE ID: B001
BIKE TYPE: Road Bike
BIKE MANUFACTURER: BMX
BIKE VALUE: 1500
STATUS: *
-----
BIKE ID: B002
BIKE TYPE: Road Bike
BIKE MANUFACTURER: Giant
BIKE VALUE: 2500
STATUS: **
-----
BIKE ID: B003
BIKE TYPE: Mountain Bike
BIKE MANUFACTURER: Raleigh
BIKE VALUE: 1000
STATUS: *
-----
BIKE ID: B004
BIKE TYPE: Kids Bike
BIKE MANUFACTURER: Canyon
BIKE VALUE: 1750
STATUS: **
-----
BIKE ID: B006
BIKE TYPE: Mountain Bike
BIKE MANUFACTURER: BMX
BIKE VALUE: 2000
STATUS: **
-----
BIKE ID: B007
BIKE TYPE: Road Bike
BIKE MANUFACTURER: Canyon
BIKE VALUE: 1800
STATUS: **
-----
BIKE ID: B008
BIKE TYPE: Hybrid Bike
BIKE MANUFACTURER: BMX
BIKE VALUE: 1500
STATUS: *
-----
```



Search



7:45 PM
11/7/2024

Question 8

PL/SQL procedure successfully completed.

BIKE_ID	BIKE_TYPE	MANUFACTURER	VALUE	STATUS
B001	Road Bike	BMX	1500	*
B002	Road Bike	Giant	2500	**
B003	Mountain Bike	Raleigh	1000	*
B004	Kids Bike	Canyon	1750	**
B006	Mountain Bike	BMX	2000	**
B007	Road Bike	Canyon	1800	**
B008	Hybrid Bike	BMX	1500	*
B009	Road Bike	Trek	1500	*
B010	Mountain Bike	Trek	2500	**
B005	Gravel Road Bike	BMX	3500	***
B011	Road Bike	BMX	2500	**

11 rows selected.

```
-- Question 8
-- Enable output display in SQL*Plus
SET SERVEROUTPUT ON;

-- Set line size and column formats for clean display in SQL*Plus
SET LINESIZE 200
COLUMN "BIKE ID" FORMAT A10
COLUMN "BIKE TYPE" FORMAT A15
COLUMN "MANUFACTURER" FORMAT A20
COLUMN "DONATION VALUE" FORMAT A15
COLUMN "STATUS" FORMAT A6;

-- Execute the query to display the results
SELECT
    b.BIKE_ID,
    b.BIKE_TYPE,
    b.MANUFACTURER,
    TO_CHAR(dn.DONATION_VALUE) AS "VALUE",
    CASE
        WHEN dn.DONATION_VALUE BETWEEN 0 AND 1500 THEN '*'
        WHEN dn.DONATION_VALUE > 1500 AND dn.DONATION_VALUE <= 3000 THEN '**'
        WHEN dn.DONATION_VALUE > 3000 THEN '***'
        ELSE 'N/A' -- Default case if needed
    END AS STATUS
FROM
    Bikes b
JOIN
    Donation dn ON b.BIKE_ID = dn.BIKE_ID;
```

Question 9

```
CREATE OR REPLACE TRIGGER prevent_donation_delete
*
ERROR at line 1:
ORA-04089: cannot create triggers on objects owned by SYS

CREATE OR REPLACE TRIGGER validate_donation_value
*
ERROR at line 1:
ORA-04089: cannot create triggers on objects owned by SYS

SQL> _
```

```
-- Question 9
-- Question 9.1
-- Create a trigger to prevent deletion from the Donation table
CREATE OR REPLACE TRIGGER prevent_donation_delete
BEFORE DELETE ON Donation
FOR EACH ROW
BEGIN
    RAISE_APPLICATION_ERROR(-20001, 'Deletion of records from the Donation table is not allowed.');
```

```
END;
/

-- Attempt to delete a record from the Donation table
DELETE FROM Donation WHERE DONATION_ID = 1; -- Replace with a valid donation ID

-- This will result in the trigger firing, and the output should be:
-- ORA-20001: Deletion of records from the Donation table is not allowed.

-- Question 9.2
-- Create a trigger to ensure valid bike value on update to the Donation table
CREATE OR REPLACE TRIGGER validate_donation_value
BEFORE UPDATE ON Donation
FOR EACH ROW
BEGIN
    -- Check if the new donation value is valid
    IF :NEW.DONATION_VALUE <= 0 THEN
        RAISE_APPLICATION_ERROR(-20002, 'Invalid donation value: Must be greater than 0.');
```

```
END IF;
END;
/

-- Attempt to update a record in the Donation table with an invalid value
UPDATE Donation
SET DONATION_VALUE = -1000
WHERE DONATION_ID = 1; -- Replace with a valid donation ID

-- This will result in the trigger firing, and the output should be:
-- ORA-20002: Invalid donation value: Must be greater than 0.
```

Question 10

Data security is paramount for organizations like BikesRUs, where sensitive information such as donor details, donation values, and bike inventories are managed. Ensuring confidentiality, integrity, and availability (CIA) of data not only protects the organization from potential data breaches but also enhances trust and compliance with regulatory requirements. This report outlines strategies and tools that will be implemented to safeguard data confidentiality and integrity at BikesRUs, along with relevant code examples.

Importance of Confidentiality

Confidentiality ensures that sensitive data is accessed only by authorized personnel. In the context of BikesRUs, this includes safeguarding donor information and financial records. To maintain confidentiality, we will implement robust access controls, encryption techniques, and user authentication mechanisms.

Tools and Platforms

1. Oracle Database Security: Oracle provides built-in security features, such as Transparent Data Encryption (TDE), which encrypts data at rest. By enabling TDE, sensitive donor information stored in the database is automatically encrypted, preventing unauthorized access.

Example:

```
ALTER TABLESPACE users  
ENCRYPTION USING 'AES256'  
DEFAULT STORAGE(ENCRYPTION);
```

2. Data Masking: Data masking tools allow sensitive information to be obfuscated in non-production environments. This technique reduces the risk of exposure while enabling developers and testers to work with realistic data without accessing sensitive details.

Example Code for Data Masking: Using Oracle Data Masking and Subsetting, you can create a masking policy:

```
BEGIN  
  DBMS_DATA_MASKING.MASK('donor_table', 'donor_id', 'masking_policy');  
END;
```

3. Implementation of User Authentication

User authentication ensures that only authorized users can access sensitive data. Strong authentication mechanisms, such as multi-factor authentication (MFA), will be implemented to enhance security.

Example:

```
CREATE USER bikes_user IDENTIFIED BY 'SecurePassword123!';  
GRANT SELECT, INSERT, UPDATE, DELETE ON Donation TO bikes_user;
```

Importance of Integrity

Data integrity guarantees that data remains accurate, consistent, and reliable over its lifecycle. Maintaining the integrity of donor records and donation values is crucial for operational efficiency and regulatory compliance.

Tools and Platforms

1. **Database Auditing:** Implementing database auditing helps monitor changes to critical data, thereby ensuring accountability. Oracle's Fine-Grained Auditing (FGA) can track who accessed or modified donor information.

Example:

```
BEGIN  
  DBMS_FGA.ADD_POLICY(  
    object_schema => 'BikesRUs',  
    object_name => 'Donation',  
    policy_name => 'donation_audit',  
    audit_condition => 'donation_value > 1500',  
    audit_column => 'donation_value'  
  );  
END;
```

2. **Checks and Constraints:** Implementing constraints in the database schema ensures that only valid data is entered. For example, a check constraint can be applied to the donation value to ensure it is greater than zero.

Example:

```
ALTER TABLE Donation  
ADD CONSTRAINT check_donation_value CHECK (DONATION_VALUE > 0);
```

Sample Code for Data Integrity:

```
CREATE OR REPLACE TRIGGER prevent_invalid_update
BEFORE UPDATE ON Donation
FOR EACH ROW
BEGIN
    IF :NEW.DONATION_VALUE <= 0 THEN
        RAISE_APPLICATION_ERROR(-20003, 'Donation value must be greater than zero.');
```

To ensure data integrity during updates, we can use triggers that validate changes to sensitive columns. For instance, the following trigger prevents updates to the donation amount if it does not meet the specified criteria:

By leveraging tools like Oracle Database Security and implementing strategies such as data masking, robust access controls, and auditing, BikesRUs can effectively safeguard sensitive information. The implementation of checks, triggers, and encryption will further enhance data integrity, providing a secure environment for donor and donation information management. The combination of these measures creates a comprehensive data security framework that protects the organization against unauthorized access and data breaches.

References (Question 10)

<https://www.oracle.com/za/security/database-security/what-is-data-security/>