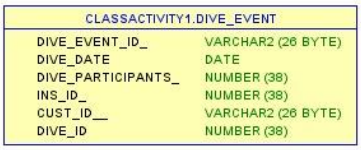
ADDB6311 PRACTICUM(EXAM)

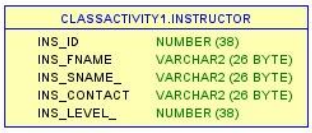
ST10131613_Lesedi_Makwakwa

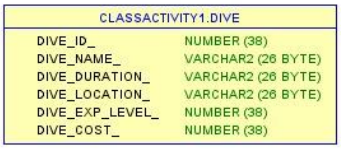
Lesedi Nkosinathi Makwakwa
St10131613@rcconnect.edu.za

Contents

-----Question 1







-----Question 2

Explanation and Assumptions

Administrator Role: The database, users, schemas, backups, and security are all under the extensive control of this role.
They are accountable for the database's overall administration and can carry out any operation within it.

General User Role: This role is more restricted and designed to carry out specific database-related tasks.
They are restricted to operations related to their assigned tasks or departmental responsibilities and have limited access rights.

1)Administrator User

Username: admin_user

Privileges:

- Full Database Access: This gives you access to all of the database's tables, including the ability to read, write, update, and delete data.
- Ability to create, edit, and delete user accounts is part of user management.
- Database schema and structure creation, modification, and deletion privileges are part of schema management.
- Reinforcement and Recuperation Authorization to perform information base reinforcements and reestablishes.

- Security Configuration: The capacity to control security settings, roles, and permissions within the database.

---Administrator Role

```
GRANT ALL PRIVILEGES ON database_name.* TO 'admin_user'@'localhost';

GRANT CREATE USER, ALTER USER, DROP USER ON database_name.* TO 'admin_user'@'localhost';

GRANT CREATE SCHEMA, ALTER SCHEMA, DROP SCHEMA ON database_name.* TO 'admin_user'@'localhost';

GRANT BACKUP_ADMIN, RELOAD ON . TO 'admin_user'@'localhost';

GRANT GRANT OPTION ON . TO 'admin_user'@'localhost';
```

General User of the System

Username: general_user

Privileges:

- Restricted Information base Access-Read and compose admittance to explicit tables pertinent to their job or office.
- Data Modification: The authority to modify and delete records from their authorized tables.
- Data Query: The capacity to execute specific queries on pertinent tables.
- Basic user management: the potential to modify their own password.

```
-- General User Role

GRANT SELECT, INSERT, UPDATE, DELETE ON database_name.table1 TO 'general_user'@'localhost';

GRANT SELECT, INSERT, UPDATE, DELETE ON database_name.table2 TO 'general_user'@'localhost';
```

----question 3

```
SELECT I.INSTRUCTOR_FNAME||', '||I.INSTRUCTOR_SNAME AS INSTRUCTOR,
CU.CUSTOMER_FNAME||', '||CU.CUSTOMER_SNAME AS CUSTOMER,
D.DIVE_LOCATION, D.DIVE_PARTICIPANT

FROM INSTRUCTOR I, CUSTOMER CU,  DIVE_LOCATION D, DIVE_PARTICIPANT D

WHERE I.INSTRUCTOR_ID = I.INSTRUCTOR_ID

AND D.DIVE_LOCATION_ID = D.DIVE_LOCATION_ID

AND CU. CUSTOMER_ID = CU. CUSTOMER_ID

AND D. DIVE_PARTICIPANT = D. DIVE_PARTICIPANT_ID
```

---Question 4

```
SET SERVEROUTPUT ON
DECLARE
    v_dive_name dives.dive_name%TYPE;
    v_dive_date dives.dive_date%TYPE;
    v_participant_count NUMBER;
BEGIN
    FOR rec IN (SELECT d.dive_name, d.dive_date,
COUNT(p.participant_id) AS participant_count
                FROM dives d
                JOIN participants p ON d.dive_id = p.dive_id
                GROUP BY d.dive_name, d.dive_date
                HAVING COUNT(p.participant_id) >= 10)
    LOOP
        v_dive_name := rec.dive_name;
        v_dive_date := rec.dive_date;
        v_participant_count := rec.participant_count;

        -- Output format
        DBMS_OUTPUT.PUT_LINE('DIVE NAME: ' || v_dive_name);
        DBMS_OUTPUT.PUT_LINE('DIVE DATE:');
        DBMS_OUTPUT.PUT_LINE(TO_CHAR(v_dive_date, 'DD/MON/YY'));
        DBMS_OUTPUT.PUT_LINE('PARTICIPANTS: ' ||
v_participant_count);
        DBMS_OUTPUT.PUT_LINE('');
    END LOOP;
END;
/
--------------------------------------------------------------------
```

----Question 5

```
DECLARE
    CURSOR dive_cursor IS
        SELECT c.first_name || ', ' || c.last_name AS customer_name,
               de.dive_name,
               COUNT(dp.customer_id) AS participant_count,
               CASE
                   WHEN COUNT(dp.customer_id) <= 4 THEN 1
                   WHEN COUNT(dp.customer_id) BETWEEN 5 AND 7 THEN 2
                   ELSE 3
               END AS instructors_required
          FROM customers c
               JOIN dive_events de ON c.customer_id = de.customer_id
               JOIN dive_participants dp ON de.event_id =
dp.event_id
         WHERE de.dive_cost > 500
         GROUP BY c.first_name, c.last_name, de.dive_name;

    v_customer_name customers.first_name%TYPE;
    v_dive_name dive_events.dive_name%TYPE;
    v_participant_count INTEGER;
    v_instructors_required INTEGER;
BEGIN
    FOR dive_rec IN dive_cursor LOOP
        v_customer_name := dive_rec.customer_name;
        v_dive_name := dive_rec.dive_name;
        v_participant_count := dive_rec.participant_count;
        v_instructors_required := dive_rec.instructors_required;

        -- Print the formatted result
        DBMS_OUTPUT.PUT_LINE('CUSTOMER: ' || v_customer_name);
        DBMS_OUTPUT.PUT_LINE('DIVE NAME: ' || v_dive_name);
```

```
        DBMS_OUTPUT.PUT_LINE('PARTICIPANTS: ' ||
v_participant_count);
        DBMS_OUTPUT.PUT_LINE('STATUS: ' || v_instructors_required ||
' instructor(s) required.');
        DBMS_OUTPUT.PUT_LINE('');
    END LOOP;
END;
/
```

----Question 6

```
CREATE OR REPLACE VIEW Vw_Dive_Event
CREATE VIEW Vw_Dive_Event AS
    SELECT
        i.instructor_id AS INS_ID,
        de.customer_id AS CUST_ID,
        c.customer_address AS CUST_ADDRESS,
        de.dive_duration AS DIVE_DURATION,
        TO_CHAR(de.dive_date, 'DD/MON/YY') AS DIVE_DATE
    FROM
        dive_events de
        JOIN instructors i ON de.instructor_id = i.instructor_id
        JOIN customers c ON de.customer_id = c.customer_id
    WHERE
        de.dive_date < TO_DATE('19-JUL-17', 'DD-MON-YY')
--------------------------------------------------------------------
```

---Question 7

```
-- Create the trigger
CREATE TRIGGER New_Dive_Event
ON dive_event
AFTER INSERT, UPDATE
AS
BEGIN
    -- Check for conditions where the number of participants is
invalid
    IF EXISTS (
        SELECT 1
        FROM inserted
        WHERE participants <= 0 OR participants > 20
    )
    BEGIN
        RAISERROR ('Number of participants must be between 1 and
20.', 16, 1);
        ROLLBACK TRANSACTION;  -- Rollback the transaction to
prevent the invalid entry
        RETURN;
    END
END;
/
---Trigger Test
-- Insert a valid entry (participants = 10)
INSERT INTO dive_event (event_name, participants)
VALUES ('Scuba Diving Competition', 10);
```

```
-- Insert an invalid entry (participants = 0)
INSERT INTO dive_event (event_name, participants)
VALUES ('Invalid Dive Event', 0);

-- Update an existing event to have more than 20 participants
UPDATE dive_event
SET participants = 25
WHERE event_name = 'Scuba Diving Competition';
```

----Question 8

```
CREATE OR REPLACE PROCEDURE CUSTOMER_Details(custid IN
customer.customer_id%TYPE)
IS
v_fname customer.customer_fname%TYPE;
v_sname customer.customer_sname%TYPE;
v_dive.dive_name%TYPE;
c_date booking.booking_date%TYPE;

Cursor info is
SELECT c.customer_fname, c.customer_sname, d.dive_name,
a.booking_date
FROM customer c
INNER JOIN bookings b ON c.customer_id = b.customer_id
INNER JOIN dives d ON b.dive_id = d.dive_id
WHERE c.customer_id = custid;
BEGIN
OPEN info;
LOOP
FETCH info INTO v_fname, v_sname,v_dive, c_date;
EXIT WHEN info%NOTFOUND;
DBMS_OUTPUT.PUT_LINE('CUSTOMER DETAILS :'||v_fname||' '||v_sname||'
booking of the current adventure '||v_dive||' on '||c_date);
END LOOP;
CLOSE info;
```

```
END;
/

--Code to test procedure

EXEC CUSTOMER_Details('cust106');

--------------------------------------------------------------------------
```

----Question 9

```
CREATE OR REPLACE FUNCTION CalculateTotalSaleAmount(Cust_ID IN
CUSTOMER.CUSTOMER_ID%TYPE)
RETURN VARCHAR2
IS
Details varchar2(100);
CURSOR C1 IS
SELECT C.CUSTOMER_FNAME||', '||C.CUSTOMER_SNAME||' Booked for a
current adventure '|| D.DIVE_NAME||' and has a new date which is '||
(BO.Booking_Date + 5)
FROM CUSTOMER C, DIVE d, BOOKING BO
WHERE
C.CUSTOMER_ID = C.CUSTOMER_ID
AND
D.DIVE_ID = D.DIVE_ID
AND C.CUSTOMER_ID = Cust_ID;

BEGIN
OPEN C1;
FETCH C1 INTO Details;

IF C1%NOTFOUND THEN
Details := 'No Booking Information found';

END IF;

RETURN Details;
CLOSE C1;

EXCEPTION WHEN OTHERS THEN
        RAISE_APPLICATION_ERROR(-20191, 'An Error was encountered -
'||SQLCODE||'-ERROR-'||SQLERRM);
END;
/

--Testing the function

SELECT CaseAdjustments('cust106')
FROM DUAL;
```

----Question 10

HTML structure:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>IT Gear Dealer Management</title>
    <link rel="stylesheet" href="styles.css"> <!-- Optional:
External CSS for styling -->
</head>
<body>
    <div class="container">
        <h1>IT Gear Dealer Management</h1>

        <div class="form-group">
            <label for="customerId">Customer ID:</label>
            <input type="text" id="customerId" placeholder="Enter
Customer ID">
        </div>

        <div class="form-group">
            <label for="diveDate">Dive Date:</label>
            <input type="date" id="diveDate">
        </div>

        <button onclick="showCustomerDetails()">Show Customer
Details</button>

        <hr>

        <div class="form-group">
            <label for="categoryId">Category ID:</label>
            <input type="text" id="categoryId" placeholder="Enter
Category ID">
        </div>

        <div class="form-group">
            <label for="adjustment">Adjustment:</label>
```

```
            <input type="number" id="adjustment" placeholder="Enter
Adjustment">
        </div>

        <button onclick="showDiveAdjustments()">Show Dive
Adjustments</button>

        <hr>

        <div id="result">
            <!-- Results will be displayed here -->
        </div>
    </div>

    <script src="script.js"></script> <!-- Include JavaScript file -
->
</body>
</html>
```

----Question 11

INTRODUCTION

Guaranteeing information and data set security is vital for ADC (the IT Stuff Vendor) to safeguard delicate client data, keep up with functional trustworthiness, and follow administrative necessities. The following are five additional approaches to improving the security of databases and data, along with their advantages and motivations:

FIVE WAYS OF ENSURING DATA AND DATABASE SECURITY.

1. Data Encryption in Transit and at Rest Strategy: Carry out encryption systems to get information both very still (put away information) and on the way (information being sent over networks).

Benefits:

Confidentiality: Even if unauthorized parties gain access to the data, encryption ensures that they will be unable to read or use it without decryption keys.

Compliance: Helps meet data protection and privacy regulations like the GDPR and HIPAA.

Risk Mitigation: Protects sensitive information like customer details and financial data, lowering the risk of data breaches and unauthorized access.

The rationale is that encryption is an essential method for protecting sensitive data from a variety of threats, such as unauthorized access and data interception during transmission.

It guarantees that regardless of whether information is compromised, it stays disjointed without the right unscrambling keys.

SECOND FACTOR

2. Vulnerability and security checks on a regular basis for databases Strategy: Direct occasional security reviews and weakness appraisals of the information base framework.

Benefits:

Identify Weaknesses: Contributes to the identification and resolution of potential security flaws and vulnerabilities in the database system.

Prevents vulnerabilities from being exploited by attackers by enabling proactive security measures. Compliance Verification: Verifies the effectiveness of security controls to ensure compliance with security policies and regulations.

Motivation: To remain proactive in the face of changing security threats, regular audits and assessments are necessary. By persistently assessing the data set foundation, ADC can improve its strength and responsiveness to arising security challenges.

THIRD AND FORTH FACTOR, BENEFITS AND METHOD.

3. Role-Based Access Control (RBAC)

Method: To enforce access controls based on the roles and responsibilities of users, implement RBAC.

4. Implement Database Activity Monitoring (DAM)

Method: Implement DAM solutions to continuously monitor and log database activities.

Benefits:

Recognition of Oddities: Helps in recognizing unapproved access endeavours, dubious exercises, and information breaks immediately.

Forensic Analysis: In the event of security incidents or compliance audits, provides comprehensive audit logs for forensic analysis. Consistence and Responsibility: Helps with meeting administrative necessities by showing adherence to security approaches and controls.

Motivation: DAM increases visibility into user and database operations, allowing for proactive security incident detection and response. By providing insight into potential threats and ensuring accountability for database access and usage, it improves ADC's security posture.

FIFTH FACTOR METHOD AND BENEFITS

5. Implement Data Masking and Redaction

Method: Use information veiling and redaction procedures to muddle delicate information components in non-creation conditions and cutoff openness underway conditions.

Benefits:

Protect Sensitive Data: Makes sure that sensitive data, like personally identifiable information, is hidden or redacted to keep it from being seen by unauthorized parties. Keep up with Information Protection: Works with consistence with information security guidelines by limiting openness of touchy information during testing and improvement exercises.

Preserve Data Utility: Protects sensitive information while allowing stakeholders to work with real-world data sets.

Motivating factors include the fact that data masking and redaction strike a balance between data usability and security, ensuring that sensitive data is safeguarded at all stages of the data lifecycle. ADC can use this strategy to support the integrity and privacy of its data without sacrificing operational effectiveness.

CONCLUSION

Executing these extra safety efforts close by isolation of obligations and trigger-based controls will essentially fortify ADC's information and data set security pose.

ADC its able to effectively mitigate risks, improve regulatory compliance, and safeguard sensitive information by implementing encryption, carrying out regular audits, implementing RBAC, deploying DAM solutions, and employing data masking and redaction techniques.

Together, these measures guarantee that ADC meets regulatory requirements, maintains customer trust, and reduces the impact of security incidents on its operations.