Unit 7. Security

7.1 Needs of security

7.2 Security and integrity violations

7.3 Access control

7.4 Authorization

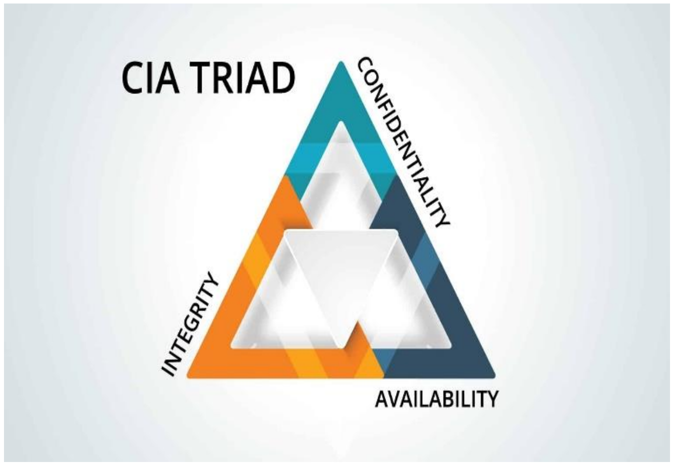
7.5 Security and Views

7.6 Encryption and decryption

https://github.com/sanjeevlcc/notes_2081/blob/main/DBMS_BIM_BSCIT_BCA/BCSIT_PU_CCT/LABS_Annapurna/Unit_7_Security.txt

Database security is the processes, tools, and controls that secure and protect databases against accidental and intentional threats. The objective of database security is to secure sensitive data and maintain the confidentiality, availability, and integrity of the database.

7.1 Needs of Security



Database security is essential to protect organizational and personal data from misuse, loss, or unauthorized access. It ensures the **CIA triad**—Confidentiality, Integrity, and Availability.

Key Objectives of Security:

1. **Confidentiality**:
   - Protect sensitive information from being disclosed to unauthorized users.
   - Examples:
     - Restricting access to employee salaries.
     - Preventing exposure of customer financial data in a retail system.
2. **Integrity**:
   - Ensure data accuracy and consistency throughout its lifecycle.
   - Examples:
     - Preventing unauthorized modifications to financial records.
     - Ensuring data input validation to avoid erroneous entries.
3. **Availability**:
   - Ensure authorized users have access to data whenever needed.
   - Examples:
     - Protecting databases from Denial of Service (DoS) attacks.
     - Maintaining backup systems to recover data during outages.

Why Security is Needed:

1. **Prevent Unauthorized Access**:
   - Protect data from hackers, malicious users, or external breaches.

  - Example: An online banking system must ensure that only account owners can view or modify their account details.

2. **Protect Sensitive Information**:
   - Safeguard personal and business-critical data.
   - Example: Encrypting healthcare data to comply with regulations like HIPAA.

3. **Regulatory Compliance**:

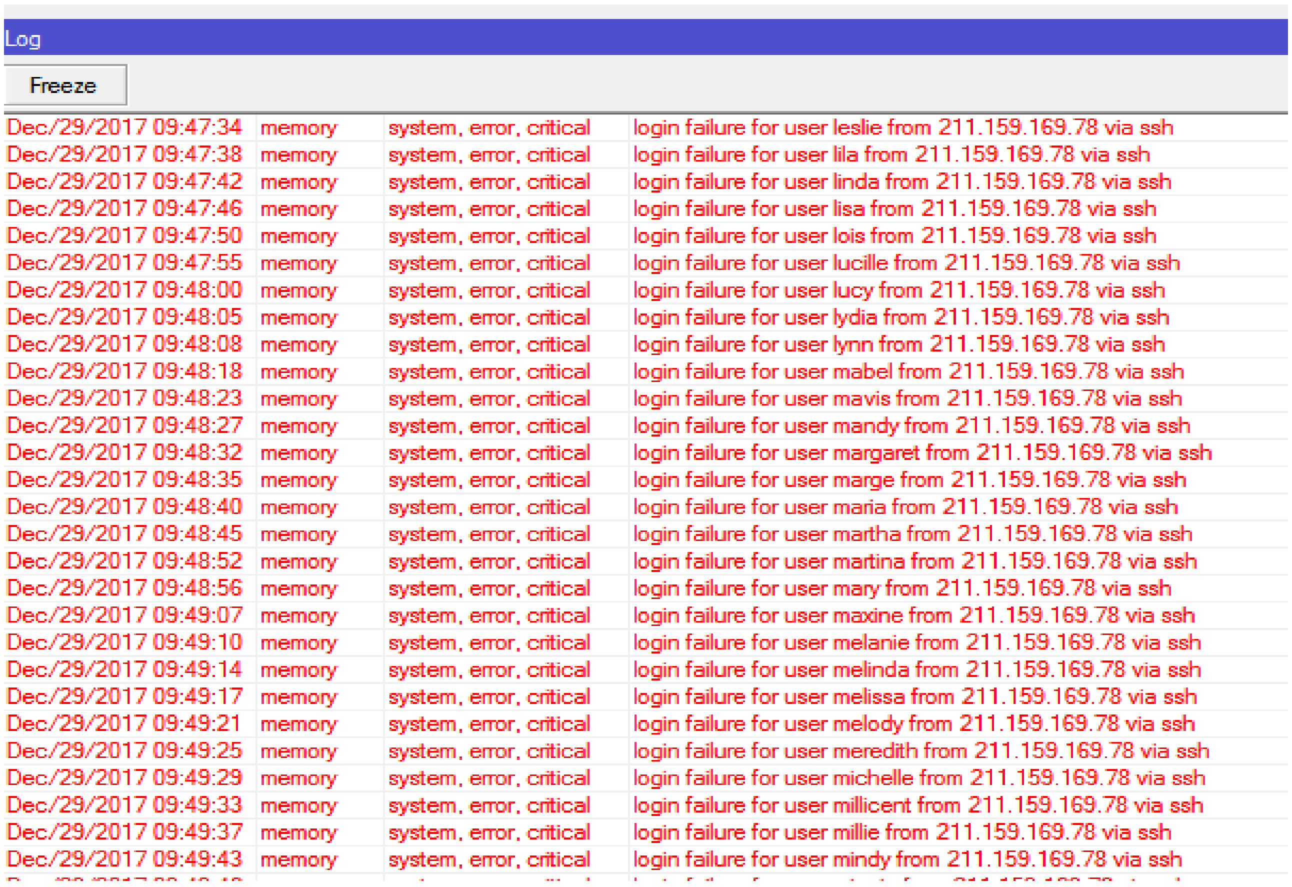
   - Adhere to data protection laws such as GDPR, CCPA, or PCI DSS.
   - Non-compliance can result in heavy penalties and loss of reputation.

4. **Defend Against Cyber Threats**:
   - Protect against threats like SQL injection, phishing attacks, and ransomware.
   - Example: A database firewall can detect and block SQL injection attempts.

5. **Support Business Continuity**:
   - Ensure the database is resilient to attacks, disasters, or failures.
   - Example: Implementing database replication and regular backups.

Example Scenario: *In an e-commerce application, the database stores customer details, orders, and payment information. Without adequate security measures:*

- Confidentiality: Hackers could steal credit card information.
- Integrity: Malicious actors could alter order details.
- Availability: A DoS attack could disrupt the system, causing downtime.



```
Dec/29/2017 09:47:34  memory  system, error, critical  login failure for user leslie from 211.159.169.78 via ssh
Dec/29/2017 09:47:38  memory  system, error, critical  login failure for user lila from 211.159.169.78 via ssh
Dec/29/2017 09:47:42  memory  system, error, critical  login failure for user linda from 211.159.169.78 via ssh
Dec/29/2017 09:47:46  memory  system, error, critical  login failure for user lisa from 211.159.169.78 via ssh
Dec/29/2017 09:47:50  memory  system, error, critical  login failure for user lois from 211.159.169.78 via ssh
Dec/29/2017 09:47:55  memory  system, error, critical  login failure for user lucille from 211.159.169.78 via ssh
Dec/29/2017 09:48:00  memory  system, error, critical  login failure for user lucy from 211.159.169.78 via ssh
Dec/29/2017 09:48:05  memory  system, error, critical  login failure for user lydia from 211.159.169.78 via ssh
Dec/29/2017 09:48:08  memory  system, error, critical  login failure for user lynn from 211.159.169.78 via ssh
Dec/29/2017 09:48:18  memory  system, error, critical  login failure for user mabel from 211.159.169.78 via ssh
Dec/29/2017 09:48:23  memory  system, error, critical  login failure for user mavis from 211.159.169.78 via ssh
Dec/29/2017 09:48:27  memory  system, error, critical  login failure for user mandy from 211.159.169.78 via ssh
Dec/29/2017 09:48:32  memory  system, error, critical  login failure for user margaret from 211.159.169.78 via ssh
Dec/29/2017 09:48:35  memory  system, error, critical  login failure for user marge from 211.159.169.78 via ssh
Dec/29/2017 09:48:40  memory  system, error, critical  login failure for user maria from 211.159.169.78 via ssh
Dec/29/2017 09:48:45  memory  system, error, critical  login failure for user martha from 211.159.169.78 via ssh
Dec/29/2017 09:48:52  memory  system, error, critical  login failure for user martina from 211.159.169.78 via ssh
Dec/29/2017 09:48:56  memory  system, error, critical  login failure for user mary from 211.159.169.78 via ssh
Dec/29/2017 09:49:07  memory  system, error, critical  login failure for user maxine from 211.159.169.78 via ssh
Dec/29/2017 09:49:10  memory  system, error, critical  login failure for user melanie from 211.159.169.78 via ssh
Dec/29/2017 09:49:14  memory  system, error, critical  login failure for user melinda from 211.159.169.78 via ssh
Dec/29/2017 09:49:17  memory  system, error, critical  login failure for user melissa from 211.159.169.78 via ssh
Dec/29/2017 09:49:21  memory  system, error, critical  login failure for user melody from 211.159.169.78 via ssh
Dec/29/2017 09:49:25  memory  system, error, critical  login failure for user meredith from 211.159.169.78 via ssh
Dec/29/2017 09:49:29  memory  system, error, critical  login failure for user michelle from 211.159.169.78 via ssh
Dec/29/2017 09:49:33  memory  system, error, critical  login failure for user millicent from 211.159.169.78 via ssh
Dec/29/2017 09:49:37  memory  system, error, critical  login failure for user millie from 211.159.169.78 via ssh
Dec/29/2017 09:49:43  memory  system, error, critical  login failure for user mindy from 211.159.169.78 via ssh
```

SQL Practice Example:

```
-- Create a user with limited access
CREATE USER LimitedUser IDENTIFIED BY 'SecurePass123';

-- Grant limited access to the Orders table
GRANT SELECT ON Orders TO LimitedUser;

-- Example of restricting access to sensitive data
CREATE VIEW PublicCustomerInfo AS
SELECT CustomerID, Name, Email FROM Customers;
```

7.2 Security and Integrity Violations

Security Violations

- **Unauthorized Access**: Accessing data without proper permissions.
- **Data Breaches**: Exfiltration of sensitive data.
- **Malware Attacks**: Using malicious software to compromise databases.

Integrity Violations

- **Data Tampering**: Unauthorized modification of data.
- **SQL Injection Attacks**: Injecting malicious SQL code to manipulate database queries or structures.
- **Inconsistent Data**: Poor validation leading to contradictory or erroneous information.

Example of SQL Injection:

-- Example of a vulnerable query

SELECT * FROM Users WHERE Username = 'admin' AND Password = '' OR '1' = '1';

-- Mitigation: Use parameterized queries

SELECT * FROM Users WHERE Username = ? AND Password = ?;

Example Scenario: An attacker uses SQL injection to gain access to sensitive data:

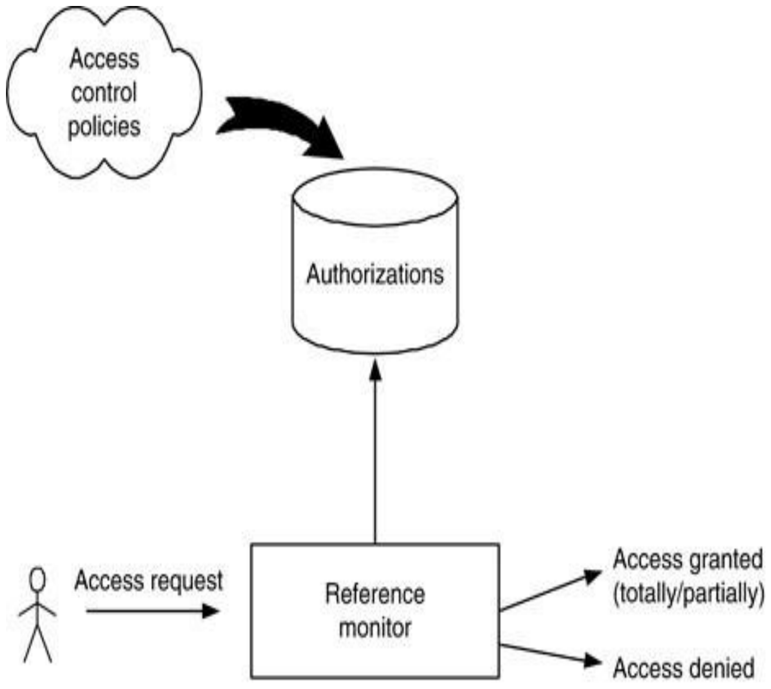
1. Sends a malicious input, e.g., ' OR '1'='1.
2. The database returns all rows from the Users table, compromising security and data integrity.

Best Practices to Prevent Violations:

1. Implement access controls to restrict unauthorized access.
2. Validate and sanitize user inputs to prevent SQL injections.
3. Regularly audit and monitor database activities to detect anomalies.

7.3 Access Control

Access control ensures that only authorized users can access specific data and perform permitted actions. It is a fundamental aspect of database security.



Types of Access Control

1. **Discretionary Access Control (DAC):**
   - Data owners specify who can access their resources and the level of access granted.
   - Example: Granting read or write permissions to specific users.
2. **Mandatory Access Control (MAC):**
   - Access is enforced by the system based on policies, such as classifications or roles.
   - Example: Only users with a "Confidential" clearance can access sensitive data.
3. **Role-Based Access Control (RBAC):**
   - Permissions are assigned to roles, and users are assigned roles.
   - Example: A "Manager" role may have access to all employee records, while a "Clerk" role may only view basic information.

SQL Implementation of Access Control

Granting and Revoking Privileges:

-- Granting SELECT and INSERT privileges to a user

GRANT SELECT, INSERT ON Employee TO User1;

-- Revoking INSERT privilege from the user

REVOKE INSERT ON Employee FROM User1;

Role-Based Access Control:

-- Creating a role for Managers

CREATE ROLE Manager;

-- Granting privileges to the Manager role

GRANT SELECT, UPDATE ON Employee TO Manager;

-- Assigning the Manager role to a user

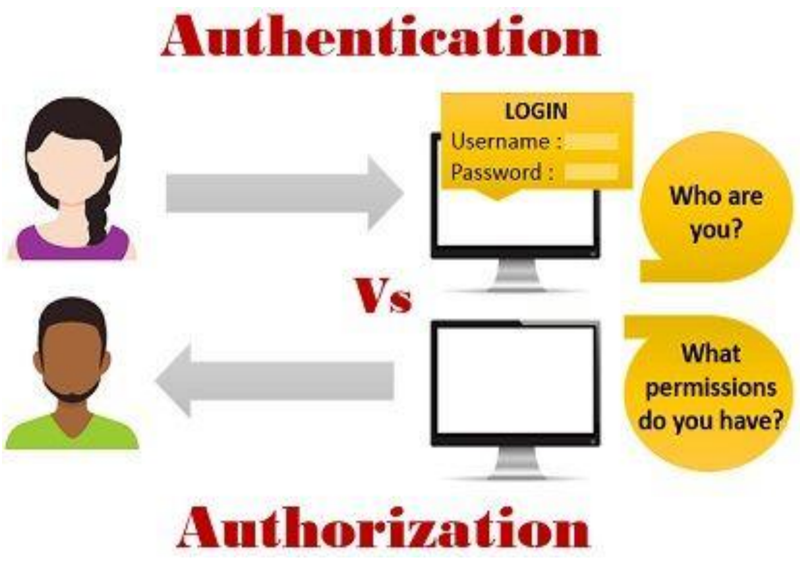
GRANT Manager TO User2;

Best Practices for Access Control

1. Follow the principle of least privilege: Grant users only the access they need.
2. Use roles to simplify permission management.
3. Regularly review and update access permissions.
4. Implement logging and monitoring to track access activities.

7.4 Authorization

Authorization determines what actions an authenticated user is allowed to perform on the database. It is crucial for enforcing security policies and ensuring data protection.



Types of Authorization

1. **Read Authorization:**
   - Allows users to view data but not modify it.
   - Example: A user can view employee names but cannot change salaries.

2. **Write Authorization:**
   - Permits users to insert, update, or delete data.
   - Example: A manager updates employee performance reviews.

3. **Execute Authorization:**
   - Grants permission to execute stored procedures or functions.
   - Example: A finance team member runs a stored procedure to calculate bonuses.

SQL Implementation of Authorization

Granting and Revoking Authorization:

-- Granting update and delete permissions to a user

GRANT UPDATE, DELETE ON Employee TO User3;

-- Revoking delete permission from a user

REVOKE DELETE ON Employee FROM User3;

Assigning Privileges via Roles:

-- Creating a role for HR personnel

CREATE ROLE HR;

-- Granting privileges to the HR role

GRANT SELECT, INSERT, UPDATE ON Employee TO HR;

-- Assigning the HR role to a user

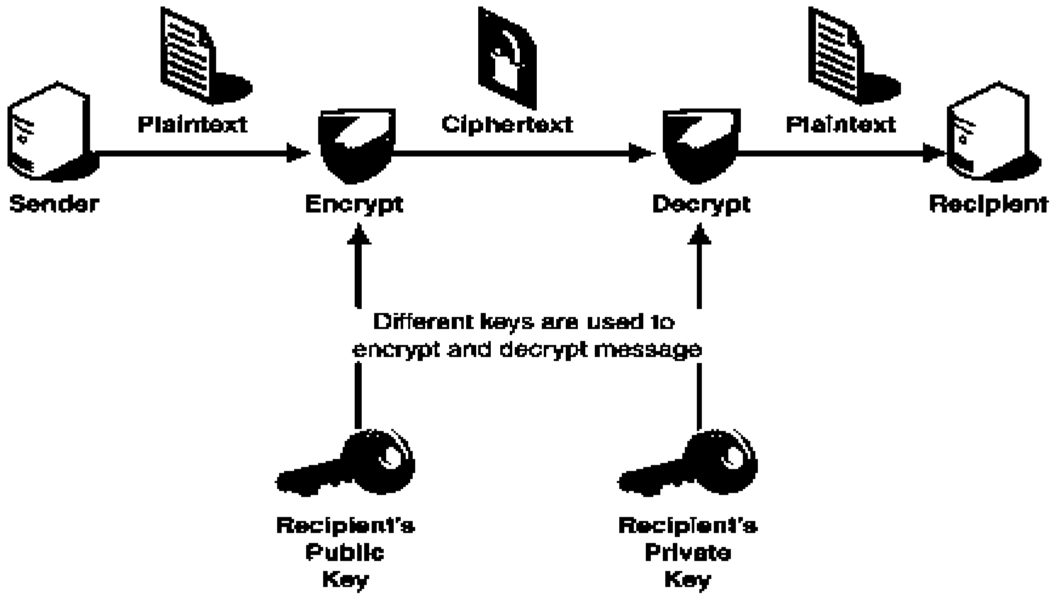
GRANT HR TO User4;

Best Practices for Authorization

1. Clearly define roles and responsibilities for all users.
2. Regularly audit user privileges to ensure compliance with security policies.
3. Combine authorization with access control mechanisms for layered security.
4. Use views to limit data exposure while granting necessary access.

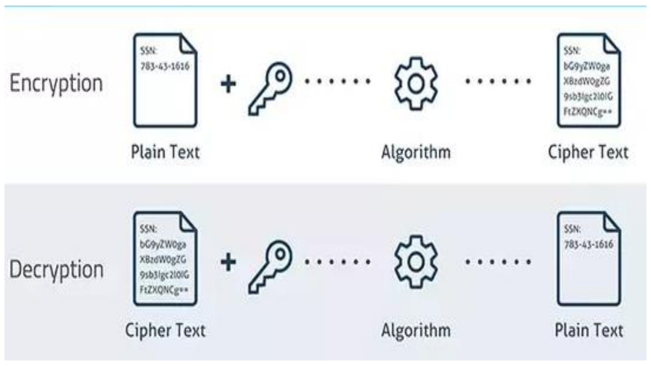
7.6 Encryption and Decryption

Encryption and decryption are fundamental techniques to protect sensitive data by converting it into unreadable formats and back to readable formats only by authorized parties.



Types of Encryption

1. **Symmetric Encryption:**
   - A single key is used for both encryption and decryption.
   - Example: AES (Advanced Encryption Standard).
2. **Asymmetric Encryption:**
   - Uses a pair of keys: one for encryption (public key) and another for decryption (private key).



  - Example: RSA (Rivest-Shamir-Adleman).

SQL Implementation of Encryption

Encrypting Data:

-- Example: Encrypt sensitive data using AES encryption

```
INSERT INTO Employee (Name, SSN)
VALUES ('John Doe', ENCRYPT('123-45-6789', 'key123'));
```

Decrypting Data:

-- Example: Decrypt sensitive data

```
SELECT Name, DECRYPT(SSN, 'key123') AS SSN
FROM Employee;
```

Benefits of Encryption

1. Protects data from unauthorized access.
2. Ensures secure data transmission.
3. Enhances compliance with regulations like GDPR and HIPAA.

Best Practices for Encryption and Decryption

1. Use strong encryption algorithms (e.g., AES-256).
2. Protect encryption keys with secure storage solutions.
3. Regularly update and rotate encryption keys.
4. Combine encryption with other security measures like access control.

Example Scenario: *A healthcare system encrypts patient records before storing them in the database. Only authorized personnel with the correct decryption keys can access the data, ensuring compliance with HIPAA regulations.*

Q/A

Fill in the Blanks (20 Questions)

1. Database security ensures _______, integrity, and availability.
2. _______ access control allows data owners to specify access levels.
3. The principle of _______ privilege grants users only the access they need.
4. SQL _______ attacks exploit vulnerabilities in database queries.
5. _______ encryption uses a single key for both encryption and decryption.
6. Asymmetric encryption involves a _______ key pair.
7. The _______ function is used in SQL to encrypt sensitive data.
8. Unauthorized data modifications are examples of _______ violations.
9. SQL _______ queries prevent injection attacks by parameterizing inputs.
10. A _______ role in RBAC simplifies permission management.
11. _______ ensures that data is accessible to authorized users when needed.
12. The _______ role in an organization might have access to all employee records.
13. _______ encryption protects data during transmission.
14. _______ is a common algorithm used in symmetric encryption.
15. The SQL command _______ grants privileges to users or roles.
16. The SQL command _______ revokes privileges from users or roles.
17. A stored procedure can be executed with _______ authorization.
18. _______ and monitoring help track access activities.
19. A _______ breach occurs when sensitive data is exfiltrated.
20. _______ encryption uses AES and RSA for strong data protection.

Multiple Choice Questions (MCQ) (20 Questions)

1. What does database security ensure?

   a) Confidentiality b) Integrity c) Availability d) All of the above

2. What type of access control is enforced by policies?

   a) DAC b) MAC c) RBAC d) None of the above

3. Which SQL command is used to grant privileges?

   a) REVOKE b) GRANT c) SELECT d) INSERT

4. What type of encryption uses a single key?

   a) Asymmetric b) Symmetric c) Public d) Private

5. SQL Injection is an example of:

   a) Data Tampering b) Unauthorized Access c) Integrity Violation d) All of the above

6. What is the purpose of the ENCRYPT function in SQL?

   a) Validate inputs b) Encrypt data c) Execute queries d) None of the above

7. What does RBAC stand for?

   a) Role-Based Access Control b) Read-Based Access Control c) Resource-Based Access Control d) None of the above

8. What is an example of symmetric encryption?

   a) RSA b) AES c) Diffie-Hellman d) All of the above

9. A _______ protects sensitive data during transmission.

   a) Firewall b) Encryption c) Backup d) Audit

10. Which key is public in asymmetric encryption?

    a) Decryption key b) Private key c) Public key d) None of the above

11. What type of authorization allows only viewing data?

    a) Write Authorization b) Read Authorization c) Execute Authorization d) None of the above

12. What prevents SQL injection attacks?

    a) Dynamic queries b) Parameterized queries

    c) Hardcoding values d) Using stored procedures

13. What role is typically granted to employees for basic data access?

a) Admin b) Manager c) Clerk d) User

14. What does MAC enforce?

a) Discretionary policies b) Mandatory policies c) Role-based policies d) None of the above

15. What is a secure way to store encryption keys?

a) In plain text b) Using secure key storage solutions c) Hardcoding in applications d) Sending via email

16. What ensures compliance with data protection laws?

a) Logging b) Encryption c) Authentication d) All of the above

17. Which SQL command removes access privileges?

a) GRANT b) REVOKE c) INSERT d) DELETE

18. Which encryption type uses a public-private key pair?

a) Symmetric b) Asymmetric c) Hashing d) None of the above

19. What is the principle of least privilege?

a) Granting all users access b) Granting only necessary permissions c) Restricting all access d) None of the above

20. What is used to detect anomalies in databases?

a) Backup b) Monitoring c) Encryption d) None of the above

Short Questions (20 Questions)

1. Define database security.
2. What are the three key objectives of database security?
3. Explain discretionary access control (DAC).
4. Describe role-based access control (RBAC).
5. What is the principle of least privilege?
6. Define SQL injection.
7. How does symmetric encryption work?
8. What is the difference between symmetric and asymmetric encryption?
9. Why is data integrity important?
10. Give an example of a security violation.

11. What is the purpose of the GRANT SQL command?

12. How do parameterized queries prevent SQL injection?

13. What is the purpose of the REVOKE SQL command?

14. Explain the role of logging in database security.

15. What is data tampering?

16. Define encryption and decryption.

17. What is an access control list (ACL)?

18. Why are roles used in RBAC?

19. What is the purpose of secure key storage?

20. Explain the importance of monitoring database activities.

Comprehensive Questions (8 Questions)

1. Discuss the needs of database security with examples.
2. Explain security and integrity violations and their prevention.
3. Describe the different types of access control and provide examples of SQL implementation.
4. Explain the concept of authorization and its types with examples.
5. Compare and contrast symmetric and asymmetric encryption with examples.
6. Write an SQL script to implement role-based access control in a database.
7. Discuss the best practices for preventing SQL inje

Answers to Fill in the Blanks

1. *Confidentiality 2. Discretionary 3. Least 4. Injection 5. Symmetric*
2. *Public-private 7. ENCRYPT 8. Integrity 9. Parameterized 10. Manager*
3. *Availability 12. Manager 13. End-to-end 14. AES 15. GRANT*
4. *REVOKE 17. Execute 18. Logging 19. Data 20. Strong*

Answers of MCQ

1. d 2. b 3. b 4. b 5. d 6. b 7. a 8. b 9. b 10. c
2. b 12. b 13. c 14. b 15. b 16. d 17. b 18. b 19. b 20. b