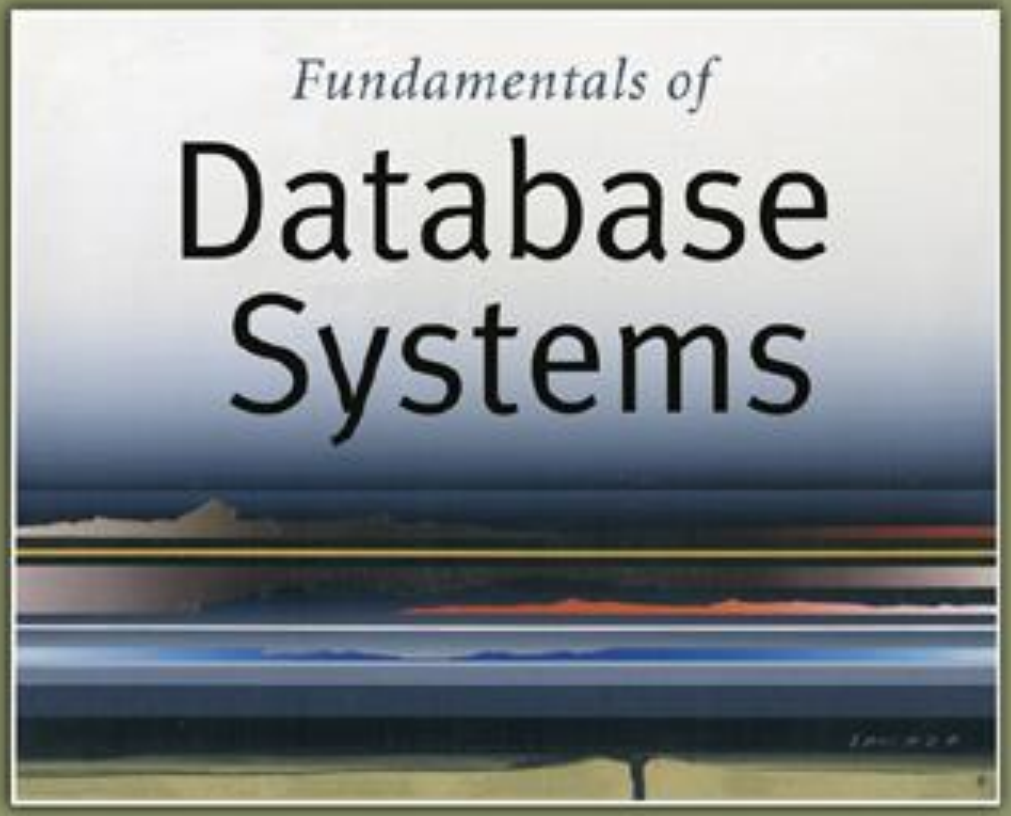
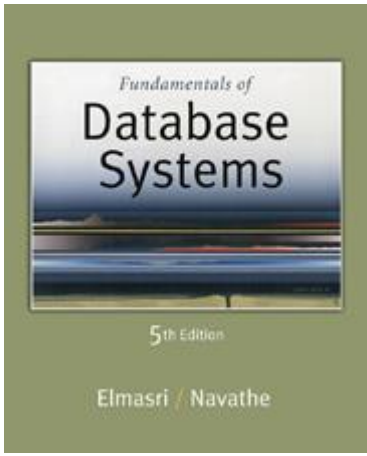


5th Edition

Elmasri / Navathe

Chapter 23

Database Security and Authorization



Chapter Outline

- 1 Database Security and Authorization
  - 1.1 Introduction to Database Security Issues
  - 1.2 Types of Security
  - 1.3 Database Security and DBA
  - 1.4 Access Protection, User Accounts, and Database Audits
- 2 Discretionary Access Control Based on Granting Revoking Privileges
  - 2.1 Types of Discretionary Privileges
  - 2.2 Specifying Privileges Using Views
  - 2.3 Revoking Privileges
  - 2.4 Propagation of Privileges Using the GRANT OPTION
  - 2.5 Specifying Limits on Propagation of Privileges

Chapter Outline (contd.)

- 3 Mandatory Access Control and Role-Based Access Control for Multilevel Security
  - 3.1 Comparing Discretionary Access Control and Mandatory Access Control
  - 3.2 Role-Based Access Control
  - 3.3 Access Control Policies for E-Commerce and the Web
- 4 Introduction to Statistical Database Security
- 5 Introduction to Flow Control
  - 5.1 Covert Channels
- 6 Encryption and Public Key Infrastructures
  - 6.1 The Data and Advanced Encryption Standards
  - 6.2 Public Key Encryption
  - 6.3 Digital Signatures

1 Introduction to Database Security Issues

Types of Security

- Legal and ethical issues, regarding the right to access certain information
- Policy issues at the governmental, institutional, or corporate level as to what kinds of information should not be made publicly available.
- System-related issues such as the system levels at which various security functions should be enforced—for example, whether a security function should be handled at the physical hardware level, the operating system level, or the DBMS level
- The need to identify multiple security levels and to categorize the data and users based on these classifications

Introduction to Database Security Issues (2)

- Threats to databases
  - Loss of **integrity**
  - Loss of **availability**
  - Loss of **confidentiality**

- To protect databases against these types of threats four kinds of countermeasures can be implemented:
  - **Access control**
  - **Inference control**
  - **Flow control**
  - **Encryption**

Introduction to Database Security Issues (3)

- In a multiuser database system, the DBMS must provide techniques to enable certain users or user groups to access selected portions of a database without gaining access to the rest of the database. This is particularly important when a large integrated database is to be used by many different users within the same organization.

- Two types of database security mechanisms:
  - **Discretionary** security mechanisms
  - **Mandatory** security mechanisms

1.2 Database Security and the DBA

- The database administrator (**DBA**) is the central authority for managing a database system.
  - The DBA's responsibilities include
    - granting privileges to users who need to use the system
    - classifying users and data in accordance with the policy of the organization
- The DBA is responsible for the overall security of the database system.

1.2 Database Security and the DBA (2)

- The DBA has a DBA account in the DBMS
  - Sometimes these are called a system or superuser account
  - These accounts provide powerful capabilities such as:
    1. Account creation
    2. Privilege granting
    3. Privilege revocation
    4. Security level assignment
  - Action 1 is access control, whereas 2 and 3 are discretionary and 4 is used to control mandatory authorization

1.3 Access Protection, User Accounts, and Database Audits

- Whenever a person or group of person s need to access a database system, the individual or group must first apply for a user account.
  - The DBA will then create a new **account id** and **password** for the user if he/she deems there is a legitimate need to access the database
- The user must log in to the DBMS by entering account id and password whenever database access is needed.

1.3 Access Protection, User Accounts, and Database Audits(2)

- The database system must also keep **track of all operations** on the database that are applied by a certain user throughout **each login session**.
  - To keep a record of all updates applied to the database and of the particular user who applied each update, we can modify **system log**, which includes an entry for each operation applied to the database that may be required for recovery from a transaction failure or system crash.

1.3 Access Protection, User Accounts, and Database Audits(3)

- If any tampering with the database is suspected, a **database audit** is performed
  - A database audit consists of reviewing the log to examine all accesses and operations applied to the database during a certain time period.
- A database log that is used mainly for security purposes is sometimes called an **audit trail**.

Discretionary Access Control Based on Granting and Revoking Privileges

- The typical method of enforcing **discretionary access control** in a database system is based on the **granting** and **revoking privileges**.

2.1Types of Discretionary Privileges

- The **account level**:
  - At this level, the DBA specifies the particular privileges that each account holds independently of the relations in the database.
- The **relation level** (or **table level**):
  - At this level, the DBA can control the privilege to access each individual relation or view in the database.

2.1Types of Discretionary Privileges(2)

- The privileges at the **account level** apply to the capabilities provided to the account itself and can include
  - the **CREATE SCHEMA** or **CREATE TABLE** privilege, to create a schema or base relation;
  - the **CREATE VIEW** privilege;
  - the **ALTER** privilege, to apply schema changes such adding or removing attributes from relations;
  - the **DROP** privilege, to delete relations or views;
  - the **MODIFY** privilege, to insert, delete, or update tuples;
  - and the **SELECT** privilege, to retrieve information from the database by using a **SELECT** query.

2.1Types of Discretionary Privileges(3)

- The second level of privileges applies to the **relation level**
  - This includes **base relations** and virtual (**view**) relations.
- The granting and revoking of privileges generally follow an authorization model for discretionary privileges known as the access matrix model where
  - The **rows** of a matrix M represents **subjects** (users, accounts, programs)
  - The **columns** represent **objects** (relations, records, columns, views, operations).
  - Each position **M(i,j)** in the matrix represents the types of privileges (read, write, update) that **subject i** holds on **object j**.

2.1Types of Discretionary Privileges(4)

- To control the granting and revoking of relation privileges, each relation R in a database is assigned and **owner account**, which is typically the account that was used when the relation was created in the first place.
  - The owner of a relation is given all privileges on that relation.
  - In SQL2, the DBA can assign and owner to a whole schema by creating the schema and associating the appropriate authorization identifier with that schema, using the **CREATE SCHEMA** command.
  - The owner account holder can **pass privileges** on any of the owned relation to other users by **granting** privileges to their accounts.

2.1Types of Discretionary Privileges(5)

- In SQL the following types of privileges can be granted on each individual relation R:
  - **SELECT** (retrieval or read) privilege on R:
    - Gives the account retrieval privilege.
    - In SQL this gives the account the privilege to use the **SELECT** statement to retrieve tuples from R.
  - **MODIFY** privileges on R:
    - This gives the account the capability to modify tuples of R.
    - In SQL this privilege is further divided into **UPDATE**, **DELETE**, and **INSERT** privileges to apply the corresponding SQL command to R.
    - In addition, both the **INSERT** and **UPDATE** privileges can specify that only certain attributes can be updated by the account.

2.1Types of Discretionary Privileges(6)

- In SQL the following types of privileges can be granted on each individual relation R (contd.):
  - **REFERENCES** privilege on R:
    - This gives the account the capability to **reference** relation R when specifying integrity constraints.
    - The privilege can also be **restricted** to specific attributes of R.

- Notice that to create a **view**, the account must have **SELECT** privilege on all relations involved in the view definition.

2.2 Specifying Privileges Using Views

- The mechanism of **views** is an important discretionary authorization mechanism in its own right. For example,
  - If the owner A of a relation R wants another account B to be able to retrieve only some fields of R, then A can create a view V of R that includes only those attributes and then grant SELECT on V to B.
  - The same applies to limiting B to retrieving only certain tuples of R; a view V' can be created by defining the view by means of a query that selects only those tuples from R that A wants to allow B to access.

2.4 Propagation of Privileges using the GRANT OPTION

- Whenever the owner A of a relation R grants a privilege on R to another account B, privilege can be given to B with or without the **GRANT OPTION**.
- If the **GRANT OPTION** is given, this means that B can also grant that privilege on R to other accounts.
  - Suppose that B is given the **GRANT OPTION** by A and that B then grants the privilege on R to a third account C, also with **GRANT OPTION**. In this way, privileges on R can **propagate** to other accounts without the knowledge of the owner of R.
  - If the owner account A now revokes the privilege granted to B, all the privileges that B propagated based on that privilege should automatically be revoked by the system.

2.5 An Example

- Suppose that the DBA creates four accounts
  - A1, A2, A3, A4
- and wants only A1 to be able to create base relations. Then the DBA must issue the following GRANT command in SQL

```
GRANT CREATETAB TO A1;
```

- In SQL2 the same effect can be accomplished by having the DBA issue a **CREATE SCHEMA** command as follows:

```
CREATE SCHAMA EXAMPLE AUTHORIZATION A1;
```

2.5 An Example(2)

- User account A1 can create tables under the schema called **EXAMPLE**.
- Suppose that A1 **creates** the two base relations **EMPLOYEE** and **DEPARTMENT**
  - A1 is then **owner** of these two relations and hence all the relation privileges on each of them.
- Suppose that A1 wants to grant A2 the privilege to insert and delete tuples in both of these relations, but A1 does not want A2 to be able to propagate these privileges to additional accounts:

```
GRANT INSERT, DELETE ON
    EMPLOYEE, DEPARTMENT TO A2;
```

2.5 An Example(3)

EMPLOYEE

Name	Ssn	Bdate	Address	Sex	Salary	Dno

DEPARTMENT

Dnumber	Dname	Mgr_ssn

Figure 23.1
Schemas for the two relations EMPLOYEE and DEPARTMENT.

2.5 An Example(4)

- Suppose that A1 wants to allow A3 to retrieve information from either of the two tables and also to be able to propagate the SELECT privilege to other accounts.
- A1 can issue the command:

```
GRANT SELECT ON EMPLOYEE, DEPARTMENT
    TO A3 WITH GRANT OPTION;
```

- A3 can grant the **SELECT** privilege on the **EMPLOYEE** relation to A4 by issuing:

```
GRANT SELECT ON EMPLOYEE TO A4;
```

  - Notice that A4 can't propagate the SELECT privilege because GRANT OPTION was not given to A4

2.5 An Example(5)

- Suppose that A1 decides to revoke the SELECT privilege on the EMPLOYEE relation from A3; A1 can issue:

```
REVOKE SELECT ON EMPLOYEE FROM A3;
```

- The DBMS must now automatically revoke the SELECT privilege on EMPLOYEE from A4, too, because A3 granted that privilege to A4 and A3 does not have the privilege any more.

2.5 An Example(6)

- Suppose that A1 wants to give back to A3 a limited capability to SELECT from the EMPLOYEE relation and wants to allow A3 to be able to propagate the privilege.
  - The limitation is to retrieve only the NAME, BDATE, and ADDRESS attributes and only for the tuples with DNO=5.
- A1 then create the view:

```
CREATE VIEW A3EMPLOYEE AS
    SELECT NAME, BDATE, ADDRESS
    FROM EMPLOYEE
    WHERE DNO = 5;
```

- After the view is created, A1 can grant **SELECT** on the view A3EMPLOYEE to A3 as follows:

```
GRANT SELECT ON A3EMPLOYEE TO A3
    WITH GRANT OPTION;
```

2.5 An Example(7)

- Finally, suppose that A1 wants to allow A4 to update only the SALARY attribute of EMPLOYEE;
- A1 can issue:

```
GRANT UPDATE ON EMPLOYEE (SALARY) TO
A4;
```

  - The **UPDATE** or **INSERT** privilege can specify particular attributes that may be updated or inserted in a relation.
  - Other privileges (**SELECT**, **DELETE**) are not attribute specific.

3 Mandatory Access Control and Role-Based Access Control for Multilevel Security

- The discretionary access control techniques of granting and revoking privileges on relations has traditionally been the main security mechanism for relational database systems.
- This is an all-or-nothing method:
  - A user either has or does not have a certain privilege.
- In many applications, and **additional security policy** is needed that classifies data and users based on security classes.
  - This approach as **mandatory access control**, would typically be **combined** with the discretionary access control mechanisms.

3 Mandatory Access Control and Role-Based Access Control for Multilevel Security(2)

- Typical **security classes** are top secret (TS), secret (S), confidential (C), and unclassified (U), where TS is the highest level and U the lowest: TS ≥ S ≥ C ≥ U

- The commonly used model for multilevel security, known as the **Bell-LaPadula model**, classifies each **subject** (user, account, program) and **object** (relation, tuple, column, view, operation) into one of the security classifications, T, S, C, or U:
  - **Clearance** (classification) of a subject S as **class(S)** and to the **classification** of an object O as **class(O)**.

3 Mandatory Access Control and Role-Based Access Control for Multilevel Security(3)

- Two restrictions are enforced on data access based on the subject/object classifications:
  - **Simple security property:** A subject S is not allowed read access to an object O unless class(S) ≥ class(O).
  - A subject S is not allowed to write an object O unless class(S) ≤ class(O). This known as the **star property** (or * property).

3 Mandatory Access Control and Role-Based Access Control for Multilevel Security(4)

- To incorporate multilevel security notions into the relational database model, it is common to consider attribute values and tuples as data objects.
- Hence, each attribute A is associated with a **classification attribute C** in the schema, and each attribute value in a tuple is associated with a corresponding security classification.
- In addition, in some models, a **tuple classification** attribute TC is added to the relation attributes to provide a classification for each tuple as a whole.
- Hence, a **multilevel relation** schema R with n attributes would be represented as
  - $R(A_1,C_1,A_2,C_2, \ldots, A_n,C_n,TC)$
- where each Ci represents the classification attribute associated with attribute A_i.

3 Mandatory Access Control and Role-Based Access Control for Multilevel Security(4)

(a) **EMPLOYEE**

Name	Salary	JobPerformance	TC
Smith U	40000 C	Fair S	S
Brown C	80000 S	Good C	S

(b) **EMPLOYEE**

Name	Salary	JobPerformance	TC
Smith U	40000 C	NULL C	C
Brown C	NULL C	Good C	C

(c) **EMPLOYEE**

Name	Salary	JobPerformance	TC
Smith U	NULL U	NULL U	U

Figure 30.2
A multilevel relation to illustrate multilevel security. (a) The original EMPLOYEE tuples. (b) Appearance of EMPLOYEE after filtering for classification C users. (c) Appearance of EMPLOYEE after filtering for classification U users. (d) Polyinstantiation of the Smith tuple.

3 Mandatory Access Control and Role-Based Access Control for Multilevel Security(7)

- In general, the **entity integrity** rule for multilevel relations states that all attributes that are members of the apparent key must not be null and must have the same security classification within each individual tuple.
- In addition, all other attribute values in the tuple must have a security classification greater than or equal to that of the apparent key.
  - This **constraint** ensures that a user can see the key if the user is permitted to see any part of the tuple at all.

3.1 Comparing Discretionary Access Control and Mandatory Access Control

- **Discretionary Access Control (DAC)** policies are characterized by a high degree of flexibility, which makes them suitable for a large variety of application domains.
  - The main drawback of **DAC** models is their vulnerability to malicious attacks, such as Trojan horses embedded in application programs.

3.2 Role-Based Access Control

- **Role-based access control (RBAC)** emerged rapidly in the 1990s as a proven technology for managing and enforcing security in large-scale enterprisewide systems.
- Its basic notion is that permissions are associated with roles, and users are assigned to appropriate roles.
- Roles can be created using the **CREATE ROLE** and **DESTROY ROLE** commands.
  - The **GRANT** and **REVOKE** commands discussed under DAC can then be used to assign and revoke privileges from roles.

4 Introduction to Statistical Database Security

- **Statistical databases** are used mainly to produce statistics on various populations.
- The database may contain **confidential data** on individuals, which should be protected from user access.
- Users are permitted to retrieve **statistical information** on the populations, such as **averages, sums, counts, maximums, minimums,** and **standard deviations**.

5 Introduction to Flow Control

- **Flow control** regulates the distribution or flow of information among accessible objects.
- A **flow** between object X and object Y occurs when a program reads values from X and writes values into Y.
  - Flow controls check that information contained in some objects does not flow explicitly or implicitly into less protected objects.
- A **flow policy** specifies the channels along which information is allowed to move.
  - The simplest flow policy specifies just two classes of information:
    - confidential (C) and nonconfidential (N)
  - and allows all flows except those from class C to class N.

5.1 Covert Channels

- A **covert channel** allows a transfer of information that violates the security or the policy.

- A **covert channel allows** information to pass from a higher classification level to a lower classification level through **improper means**.

6 Encryption and Public Key Infrastructures

- **Encryption** is a means of maintaining secure data in an insecure environment.
- **Encryption** consists of applying an **encryption algorithm** to data using some prespecified **encryption key**.
- The resulting data has to be **decrypted** using a **decryption key** to recover the original data.

6.1 The Data and Advanced Encryption Standards

- The **Data Encryption Standard (DES)** is a system developed by the U.S. government for use by the general public.
  - It has been widely accepted as a cryptographic standard both in the United States and abroad.
- **DES** can provide end-to-end encryption on the channel between the sender A and receiver B.

6.1 The Data and Advanced Encryption Standards(2)

- **DES** algorithm is a careful and complex combination of two of the fundamental building blocks of encryption:
  - **substitution** and **permutation** (transposition).
- The **DES** algorithm derives its strength from repeated application of these two techniques for a total of 16 cycles.
  - **Plaintext** (the original form of the message) is **encrypted** as blocks of **64 bits**.

6.1 The Data and Advanced Encryption Standards(3)

- After questioning the adequacy of **DES**, the National Institute of Standards (**NIST**) introduced the Advanced Encryption Standards (**AES**).
  - This algorithm has a block size of **128 bits** and thus takes longer time to crack.

6.2 Public Key Encryption

- In 1976 Diffie and Hellman proposed a new kind of cryptosystem, which they called **public key encryption**.
- **Public key algorithms** are based on **mathematical functions** rather than operations on bit patterns.
  - They also involve the use of **two separate keys**
    - in contrast to conventional encryption, which uses only one key.
  - The use of two keys can have profound consequences in the areas of confidentiality, key distribution, and authentication.

6.2 Public Key Encryption(2)

- The two keys used for public key encryption are referred to as the **public key** and the **private key**.
  - the **private key** is kept secret, but it is referred to as private key rather than a secret key (the word used in conventional encryption to avoid confusion with conventional encryption).

Summary

- 1 Database Security and Authorization
- 2 Discretionary Access Control
- 3 Mandatory Access Control and Role-Based Access Control for Multilevel Security
- 4 Statistical Database Security
- 5 Flow Control
- 6 Encryption and Public Key Infrastructures