Advanced Data Management Techniques

Module 2

IT – TE – ADMT(DLO)

Prof. Seema S. Redekar
Assistant Professor
Dept. of Information Technology,
SIES Graduate School of Technology

SIES Graduate School of Technology
RISE WITH EDUCATION

Learning Objectives

Lecture No:11

1. To explain Database Security issues.
2. To explain the terms like Access Protection, User Accounts, and Database Audits.

Introduction to Database Security Issues

- Types of Security
- Database Security addresses many issues:
  - Various legal and ethical issues regarding the right to access certain information
  - Policy issues at the governmental, institutional or corporate level
  - System related issues
  - To identify multiple security levels and to categorize the data and users

Introduction to Database Security Issues

- 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

- A DBMS typically includes a database security and authorization subsystem that is responsible for ensuring the security portions of a database against unauthorized access.
- Two types of database security mechanisms:
  - **Discretionary** security mechanisms
  - **Mandatory** security mechanisms
- The security mechanism of a DBMS must include provisions for restricting access to the database as a whole
  - This function is called **access control** and is handled by creating user accounts and passwords to control login process by the DBMS.

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.
- The DBA has a DBA account in the DBMS (system or super user account)
  1. Account creation
  2. Privilege granting
  3. Privilege revocation
  4. Security level assignment

Access Protection and User Accounts

- 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 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, **system log is maintained**, which includes an entry for each operation applied to the database that may be required for recovery from a transaction failure or system crash.

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**.
- Types 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.

Types of Discretionary Privileges

- 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.

Types of Discretionary Privileges

- 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**.

Types of Discretionary Privileges

		OBJECTS			
		File 1	File 2	File 3	File 4
SUBJECTS	User A	Own Read Write		Own Read Write	
	User B	Read	Own Read Write	Write	Read
	User C	Read Write	Read		Own Read Write

Types of Discretionary Privileges

- 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.

Types of Discretionary Privileges

- 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.

Types of Discretionary Privileges

- 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.

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.

Revoking Privileges

- In some cases it is desirable to grant a privilege to a user temporarily. For example,
  - The owner of a relation may want to grant the **SELECT** privilege to a user for a specific task and then revoke that privilege once the task is completed.
  - Hence, a mechanism for **revoking** privileges is needed. In SQL, a **REVOKE** command is included for the purpose of **canceling privileges**.

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.

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;

Example

- 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;**

Example

EMPLOYEE

Name	Ssn	Bdate	Address	Sex	Salary	Dno

DEPARTMENT

Dnumber	Dname	Mgr_ssn

Figure 23.1
Schemas for the two relations EMPLOYEE and DEPARTMENT.

Example

- 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

Example

- 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.

Example

- 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;
```

Example

- 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.

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.

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

- 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)**.

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

- 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).

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.

Role-Based Access Control

- **RBAC** appears to be a viable alternative to traditional discretionary and mandatory access controls; it ensures that only authorized users are given access to certain data or resources.
- Many DBMSs have allowed the concept of roles, where privileges can be assigned to roles.
- Role hierarchy in **RBAC** is a natural way of organizing roles to reflect the organization's lines of authority and responsibility.

Role-Based Access Control

- Another important consideration in **RBAC** systems is the possible temporal constraints that may exist on roles, such as time and duration of role activations, and timed triggering of a role by an activation of another role.
- Using an **RBAC** model is highly desirable goal for addressing the key security requirements of Web-based applications.
- In contrast, discretionary access control (**DAC**) and mandatory access control (**MAC**) models **lack capabilities** needed to support the security requirements emerging enterprises and Web-based applications.

Learning Objectives

Lecture No. 14

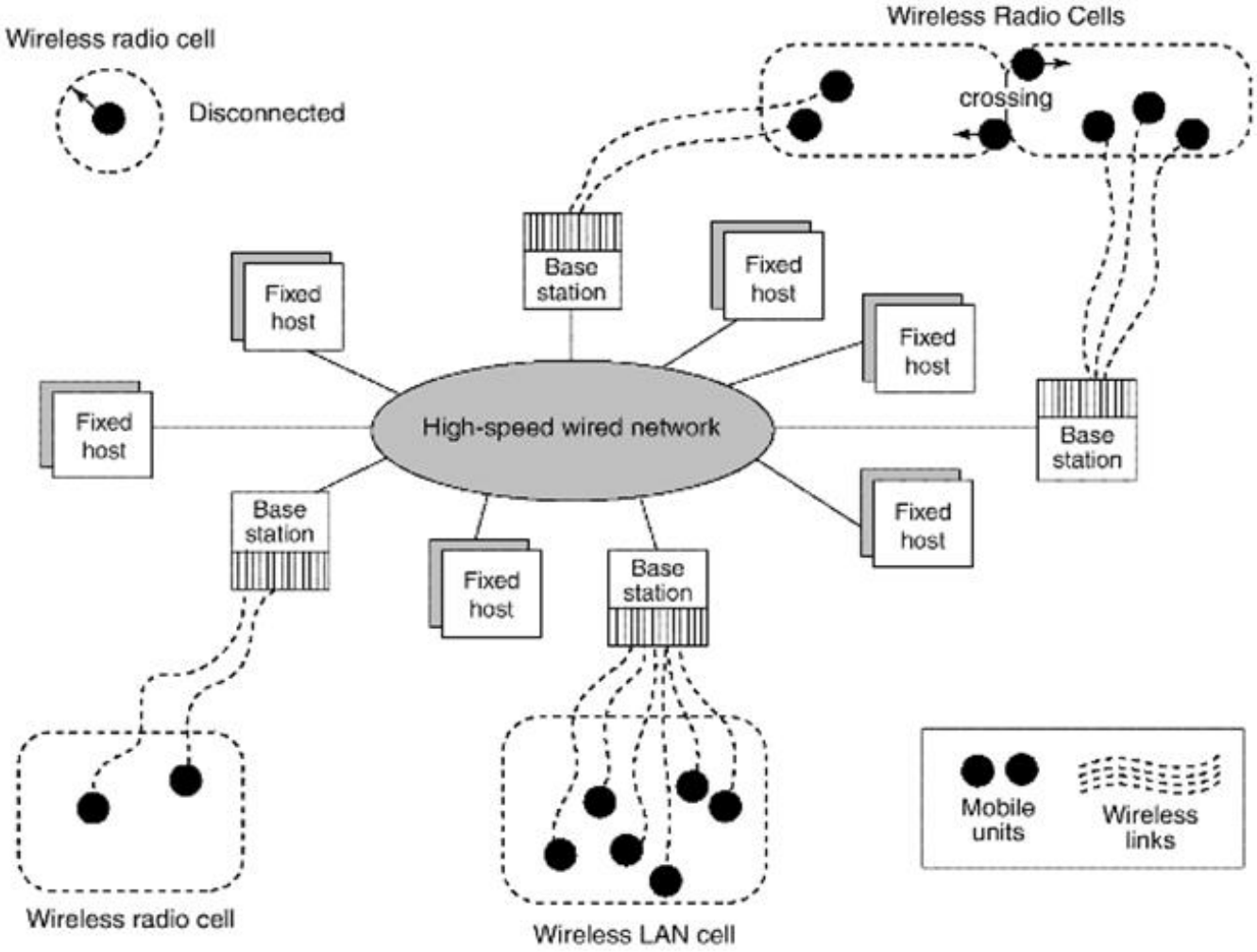
1. To explain advanced Database Models like Mobile databases, Temporal databases, Spatial databases.

Mobile Databases

- Recent advances in portable and wireless technology led to mobile computing, a new dimension in data communication and processing.
- Portable computing devices coupled with wireless communications allow clients to access data from virtually anywhere and at any time.
- There are a number of hardware and software problems that must be resolved before the capabilities of mobile computing can be fully utilized.
- Some of the software problems – which may involve data management, transaction management, and database recovery – have their origins in distributed database systems.

Mobile Databases

- In mobile computing, the problems are more difficult, mainly:
  - The limited and intermittent connectivity afforded by wireless communications.
  - The limited life of the power supply(battery).
  - The changing topology of the network.
  - In addition, mobile computing introduces new architectural possibilities and challenges



Mobile Computing Architecture

Characteristics of Mobile Environments

- The characteristics of mobile computing include:
  - **Communication latency**
  - **Intermittent connectivity**
  - **Limited battery life**
  - **Changing client location**

Data Management Issues

- From a data management standpoint, mobile computing may be considered a variation of distributed computing. Mobile databases can be distributed under two possible scenarios:
- **The entire database is distributed mainly among the wired components, possibly with full or partial replication.**
    - A base station or fixed host manages its own database with a DBMS-like functionality, with additional functionality for locating mobile units and additional query and transaction management features to meet the requirements of mobile environments.
- **The database is distributed among wired and wireless components.**
    - Data management responsibility is shared among base stations or fixed hosts and mobile units.

Data Management Issues

- Data management issues as it is applied to mobile databases:
  - Data distribution and replication
  - Transactions models
  - Query processing
  - Recovery and fault tolerance
  - Mobile database design
  - Location-based service
  - Division of labor
  - Security

Temporal Database

- Time Representation, Calendars, and Time Dimensions
- Time is considered **ordered sequence** of **points** in some **granularity**
- A **calendar** organizes time into different time units for convenience.
- Point events
  - Single time point event
    - E.g., bank deposit
  - Series of point events can form a time series data
- Duration events
  - Associated with specific time period
  - Time period is represented by start time and end time

Temporal Database Concepts

- Transaction time
  - The time when the information from a certain transaction becomes valid
- Bitemporal database
  - Databases dealing with two time dimensions
- Incorporating Time in Relational Databases Using Tuple Versioning
  - Add to every tuple
    - Valid start time
    - Valid end time

Figure 24.7
Different types of temporal relational databases. (a) Valid time database schema. (b) Transaction time database schema. (c) Bitemporal database schema.

Figure 24.8
Some tuple versions in the valid time relations EMP_VT and DEPT_VT.

EMP_VT

Name	Ssn	Salary	Dno	Supervisor_ssn	Vst	Vet
Smith	123456789	25000	5	333445555	2002-06-15	2003-05-31
Smith	123456789	30000	5	333445555	2003-06-01	Now
Wong	333445555	25000	4	999887777	1999-08-20	2001-01-31
Wong	333445555	30000	5	999887777	2001-02-01	2002-03-31
Wong	333445555	40000	5	888665555	2002-04-01	Now
Brown	222447777	28000	4	999887777	2001-05-01	2002-08-10
Narayan	666884444	38000	5	333445555	2003-08-01	Now

. . .

DEPT_VT

Dname	Dno	Manager_ssn	Vst	Vet
Research	5	888665555	2001-09-20	2002-03-31
Research	5	333445555	2002-04-01	Now

Temporal Database Concepts

- Incorporating Time in Object-Oriented Databases Using Attribute Versioning
- A single complex object stores all temporal changes of the object
- **Time varying attribute**
  - An attribute that changes over time
  - E.g., age
- **Non-Time varying attribute**
  - An attribute that does **not** changes over time
  - E.g., date of birth

Spatial Databases

- Keep track of objects in a multi-dimensional space
  - Maps
  - Geographical Information Systems (**GIS**)
  - Weather
- In general spatial databases are n-dimensional
- Typical Spatial Queries
- **Range** query: Finds objects of a particular type within a particular distance from a given location
  - E.g., India Gate in Delhi, India
- Nearest Neighbor query: Finds objects of a particular type that is nearest to a given location
  - E.g., Nearest to India gate , India

MCQs

Q. Grant and revoke are ..... statements.

A. DDL

B. TCL

C. DCL

D. DML

Q. SQL Authorization mechanism doesn't grants privileges on---

A. Specified Attribute

B. Specified tuples

C. Entire relation

D. None of the above

MCQs

Q. Which of the following is the time of temporal data that record when a fact was recorded in a database?

A. Valid Time

B. Transaction Time

C. Exit Time

D. Enter Time

Q. Most _____ allow the representation of simple geometric objects such as points, lines and polygons.

A) Active database

B) temporal database

C) spatial database

D) deductive databases

MCQs

Q. By *'spatial data'* we mean data that has

A. Complex values

B. Positional values

C. Graphic values

D. Decimal values

Q. Mobile Computing allows transmission of data from one wireless-enabled device to another_

A. Any device

B. Wired device

C. Wireless-enabled device

D. one of the above

Thank You!

(seemasr@sies.edu.in)