

## Lecture 10: Security

- **Security & Authorization:**  
SQL Access Controls

## Security and Authorization

- **Objectives:**
  - **Secrecy:** Users should not be able to see things they are not supposed to.
    - E.g., A student can't see other students' grades.
  - **Integrity:** Users should not be able to modify things they are not supposed to.
    - E.g., Only instructors can assign grades.
  - **Availability:** Users should be able to see and modify things they are allowed to.
- **OS, Network security issues also relevant**
  - Only database issues discussed here

## Access Controls

- A **security policy** specifies who is authorized to do what.
- A **security mechanism** allows us to enforce a chosen security policy.
- Two main mechanisms at the DBMS level:
  - Discretionary access control
  - Mandatory access control

## Discretionary Access Control

- Based on the concept of access rights or **privileges** for objects (tables and views), and mechanisms for giving users privileges (and revoking privileges).
- Creator of a table automatically gets all privileges on it.
  - DBMS tracks who subsequently gains and loses privileges, and only allows requests from users who have the necessary privileges.

## GRANT Command

`GRANT privileges ON object TO auth [WITH GRANT OPTION]`

- The following **privileges** can be specified:
  - ❖ **SELECT**: Can read all columns.
  - ❖ **INSERT**: Can insert tuples.
  - ❖ **DELETE**: Can delete tuples.
  - ❖ **REFERENCES (col-name)**: Can define foreign keys (in other tables) that refer to this column.
- **Auth** is a group of users.
- If a user has a privilege with the **GRANT OPTION**, can pass privilege on to other users (with or without passing on the **GRANT OPTION**).
- Only owner can execute **CREATE**, **ALTER**, and **DROP**.

CS386/586 Introduction to Database Systems, © Lois Delcambre, David Maier 1999-2011  
Some slides adapted from R. Ramakrishnan, with permission

5

## REVOKE Command

`REVOKE privileges ON object FROM auth [RESTRICT|CASCADE]`

- Must be issued by a user who granted "privileges" on "object" to "auth".
- **CASCADE**:
  - *abandon* anyone granted this privilege by auth, and so on recursively.
  - If an auth's privilege is abandoned and this is the last possible abandonment, then revoke auth's privilege
  - "and so on recursively": Abandon anyone granted the privilege by the auth just abandoned.
- **RESTRICT**: If **CASCADE** would cause anyone else to be abandoned, then cancel the command.

CS386/586 Introduction to Database Systems, © Lois Delcambre, David Maier 1999-2011  
Some slides adapted from R. Ramakrishnan, with permission

6

## Examples of Grant, Revoke

- Suppose Len has created the agent table, then these commands are executed in this order
  - Len: GRANT UPDATE, SELECT ON agents TO Yingjin
  - Len: GRANT UPDATE ON agents TO Parisa WITH GRANT OPTION
  - Parisa: GRANT UPDATE ON agents TO Yingjin
  - Len: REVOKE UPDATE ON agents FROM Parisa CASCADE
  - Len: REVOKE SELECT ON agents FROM Yingjin CASCADE
- What privileges does Parisa have on agents?
- What privileges does Yingjin have on agents?
- Can Yingjin issue UPDATE agents A SET A.age=25?
- Can Yingjin issue UPDATE agents A SET A.age=A.age+1?

CS386/586 Introduction to Database Systems, © Lois Delcambre, David Maier 1999-2011  
Some slides adapted from R. Ramakrishnan, with permission

7

## REVOKE more precisely

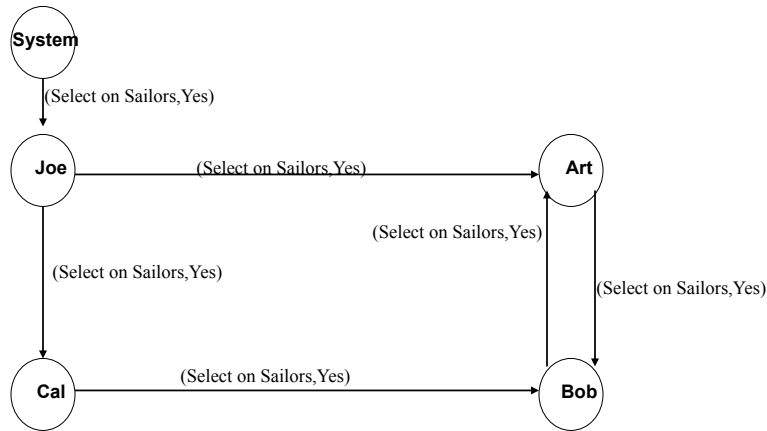
### Steps to be taken after "REVOKE P ON O FROM A"

- First consider the case CASCADE
  1. Draw an authorization graph of current state (next slide)
  2. Delete the arc corresponding to "GRANT P ON O TO A"
  3. Also delete any arc not accessible from SYSTEM via grants of the privilege P on O
  4. If an auth has no incoming arcs granting P on O, then auth loses the privilege P on O.
- RESTRICT is the same as before: If CASCADE would cause any arc other than "GRANT ON P TO A" to be deleted, then abort the command.

CS386/586 Introduction to Database Systems, © Lois Delcambre, David Maier 1999-2011  
Some slides adapted from R. Ramakrishnan, with permission

8

## Authorization Graph



CS386/586 Introduction to Database Systems, © Lois Delcambre, David Maier 1999-2011  
Some slides adapted from R. Ramakrishnan, with permission

9

## Role-Based Authorization

- In SQL-92, privileges are actually assigned to **authorization ids**, which can denote a single user or a group of users.
- In SQL:1999 privileges are assigned to **roles**.
  - Roles can then be granted to users and to other roles.
  - Reflects how real organizations work.

CS386/586 Introduction to Database Systems, © Lois Delcambre, David Maier 1999-2011  
Some slides adapted from R. Ramakrishnan, with permission

10