CS2102 Database Systems

Database Security

NUS National University of Singapore | School of Computing

Leading The World With Asia's Best

Objectives

- Secrecy: Information should not be disclosed to unauthorized users.
  - E.g., A student should not be allowed to see other students' grades.
- Integrity: Only authorized users should be allowed to modify data.
  - E.g., Only instructors can change grades.
- Availability: Authorized users should not be denied access.

Access Controls

- A security policy specifies who is authorized to do what.
- A security mechanism allows us to enforce a chosen security policy.
- Views
  - Create a "window" on a collection of data that is appropriate for some group of users.
  - Limit access to sensitive data by providing access to a restricted version of that data

Access Controls

- Two main approaches to access control in DBMS:
  - Discretionary Access Control (DAC)
    - Based on the concept of access rights for tables and views.
  - Mandatory Access Control (MAC)
    - Based on system-wide policies that cannot be changed by individual users.

Discretionary Access Control

- Access rights or privileges for objects (tables and views)
- Mechanisms for giving users privileges (and revoking privileges).
  - GRANT and REVOKE commands in SQL
- Creator of a table or a view automatically gets all privileges on it.
- DMBS keeps track of how these privileges are granted to other users, and possibly revoked, and ensures that at all times only users with the necessary privileges can access an object.

GRANT Command

GRANT privilege ON object TO users [WITH GRANT OPTION]

- The following privilege can be specified:
  - **SELECT**: Can read all columns (including those added later via ALTER TABLE command).
  - **INSERT (col-name):** Can insert tuples with non-null or non-default values in this column.
    - **INSERT**: same right to all columns.
  - **DELETE**: Can delete tuples.
  - **REFERENCES (col-name):** Can define foreign keys (in other tables) that refer to this column.
    - **REFERENCES**: same right to all columns

GRANT Command

GRANT privilege ON object TO users [WITH GRANT OPTION]

- If a user has privilege with GRANT OPTION, he can pass the privilege on to other users (with or without passing on the GRANT OPTION).
- Only the owner can execute CREATE, ALTER, and DROP.

Example Schema

- Suppose user Joe created the following tables:

```
Sailors(sid:integer,sname:string,
        rating:integer, age:real)

Boats(bid:integer,bname:string,color:string)

Reserves(sid:integer,bid:integer,day:dates)
```

Example View

```
CREATE VIEW ActiveSailors(name,age,day) AS
     SELECT S.sname, S.age, R.day
     FROM Sailors S, Reserves R
     WHERE S.sid = R.sid AND S.rating > 6
```

- A user who can access ActiveSailors, but not Sailors or Reserves, knows the names of sailors who have reservation but cannot find out the bids of boats reserved by a given sailor.

GRANT and REVOKE of Privileges

- Joe can execute following GRANT commands:
  - GRANT INSERT, SELECT ON Sailors TO Horatio
    - Horatio can query Sailors or insert tuples into it.
  - GRANT DELETE ON Sailors TO Yuppy WITH GRANT OPTION
    - Yuppy can delete tuples, and also authorize others to do so.
  - GRANT UPDATE (*rating*) ON Sailors TO Dustin
    - Dustin can update (only) the *rating* field of Sailors tuples.
  - GRANT SELECT ON ActiveSailors TO Guppy, Yuppy
    - This does NOT allow the 'uppies to query Sailors directly!

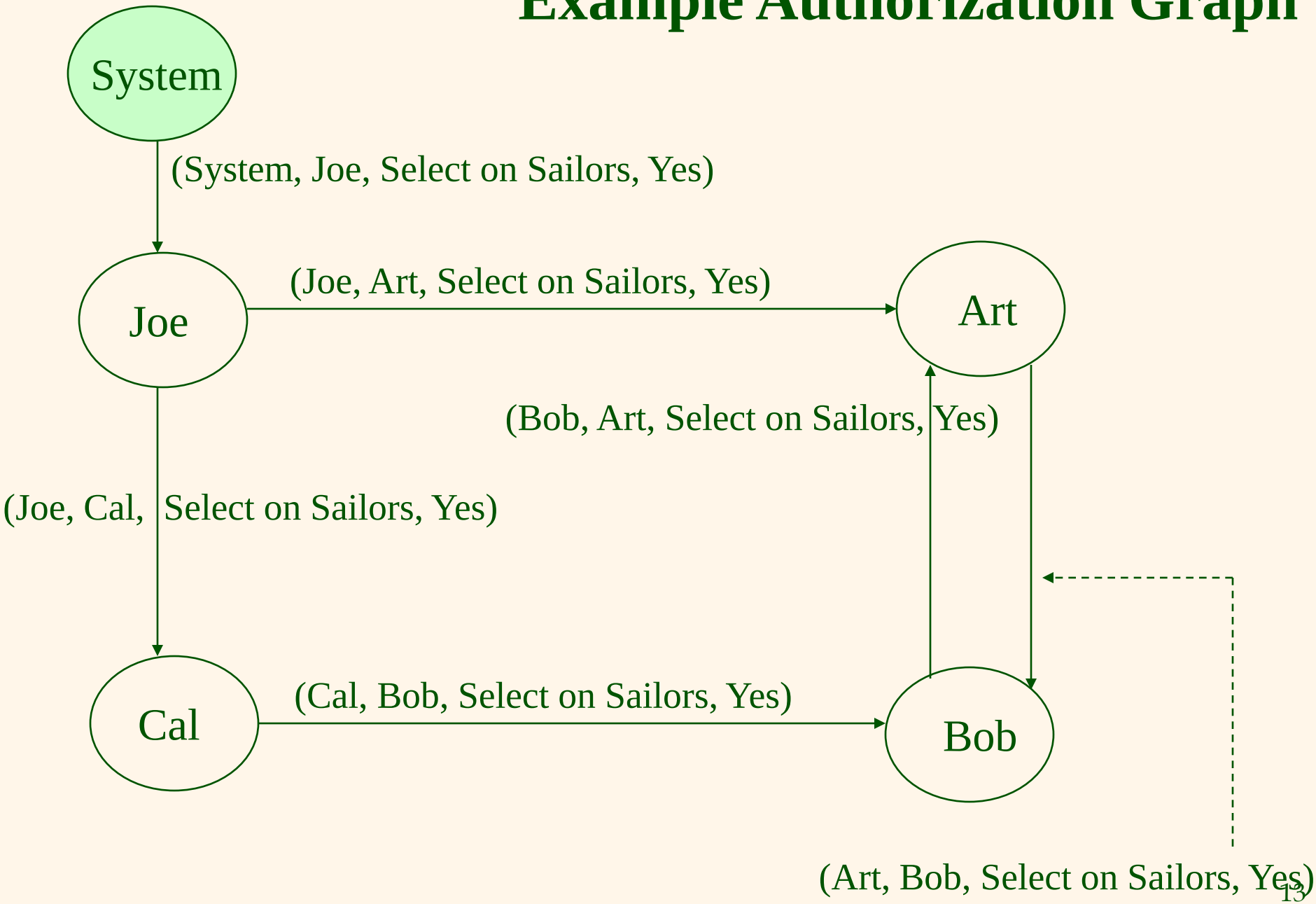
GRANT and REVOKE of Privileges

- REVOKE: When a privilege is revoked from X, it is also revoked from all users who got it *solely* from X.
- Authorization Graph
  - Nodes are users
  - Edges indicate how privileges are passed

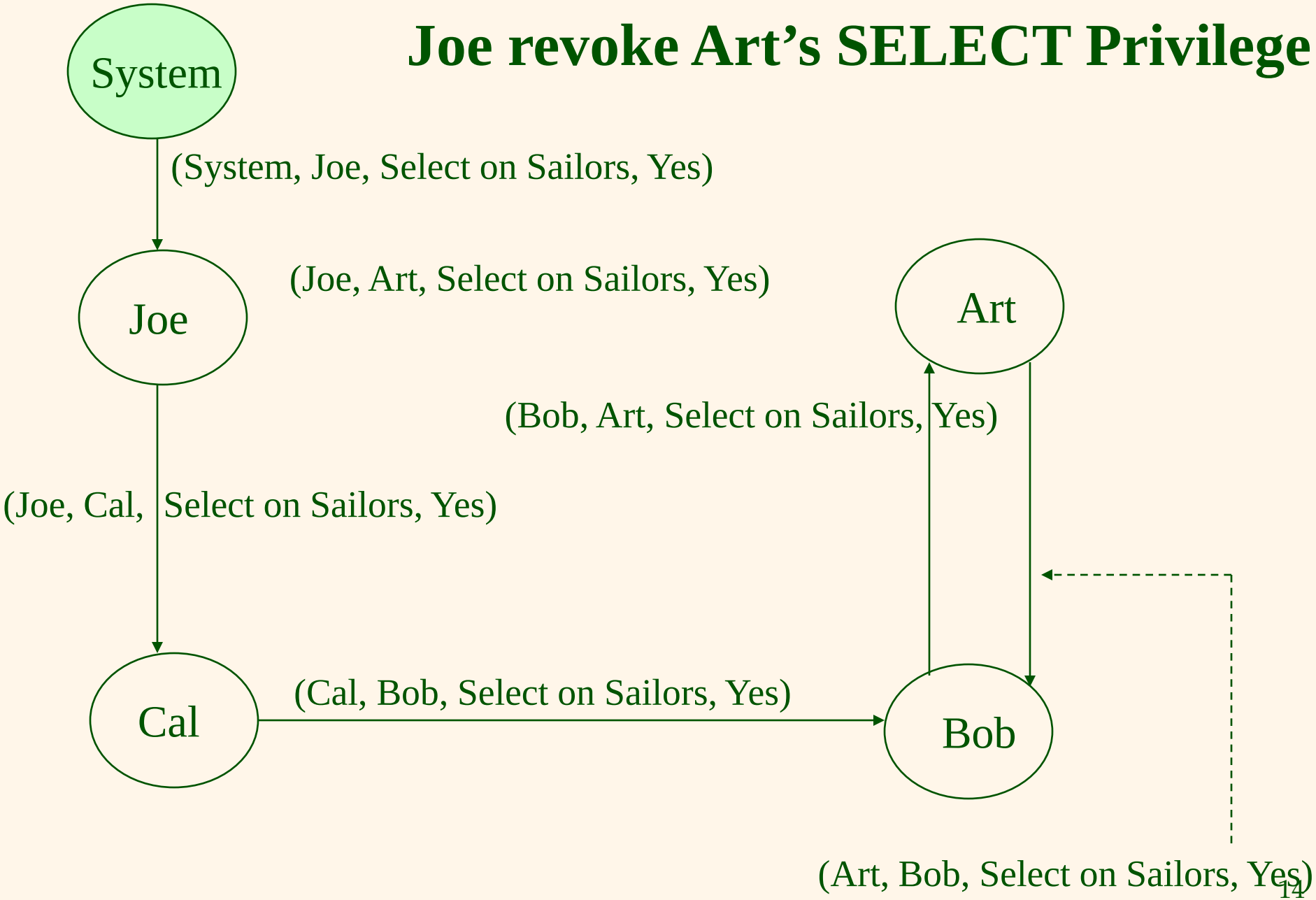
GRANT and REVOKE of Privileges

- GRANT SELECT ON Sailors TO Art WITH GRANT OPTION (executed by Joe)
- GRANT SELECT ON Sailors TO Bob WITH GRANT OPTION (executed by Art)
- GRANT SELECT ON Sailors TO Art WITH GRANT OPTION (executed by Bob)
- GRANT SELECT ON Sailors TO Cal WITH GRANT OPTION (executed by Joe)
- GRANT SELECT ON Sailors TO Bob WITH GRANT OPTION (executed by Cal)
- REVOKE SELECT ON Sailors FROM Art CASCADE (executed by Joe)

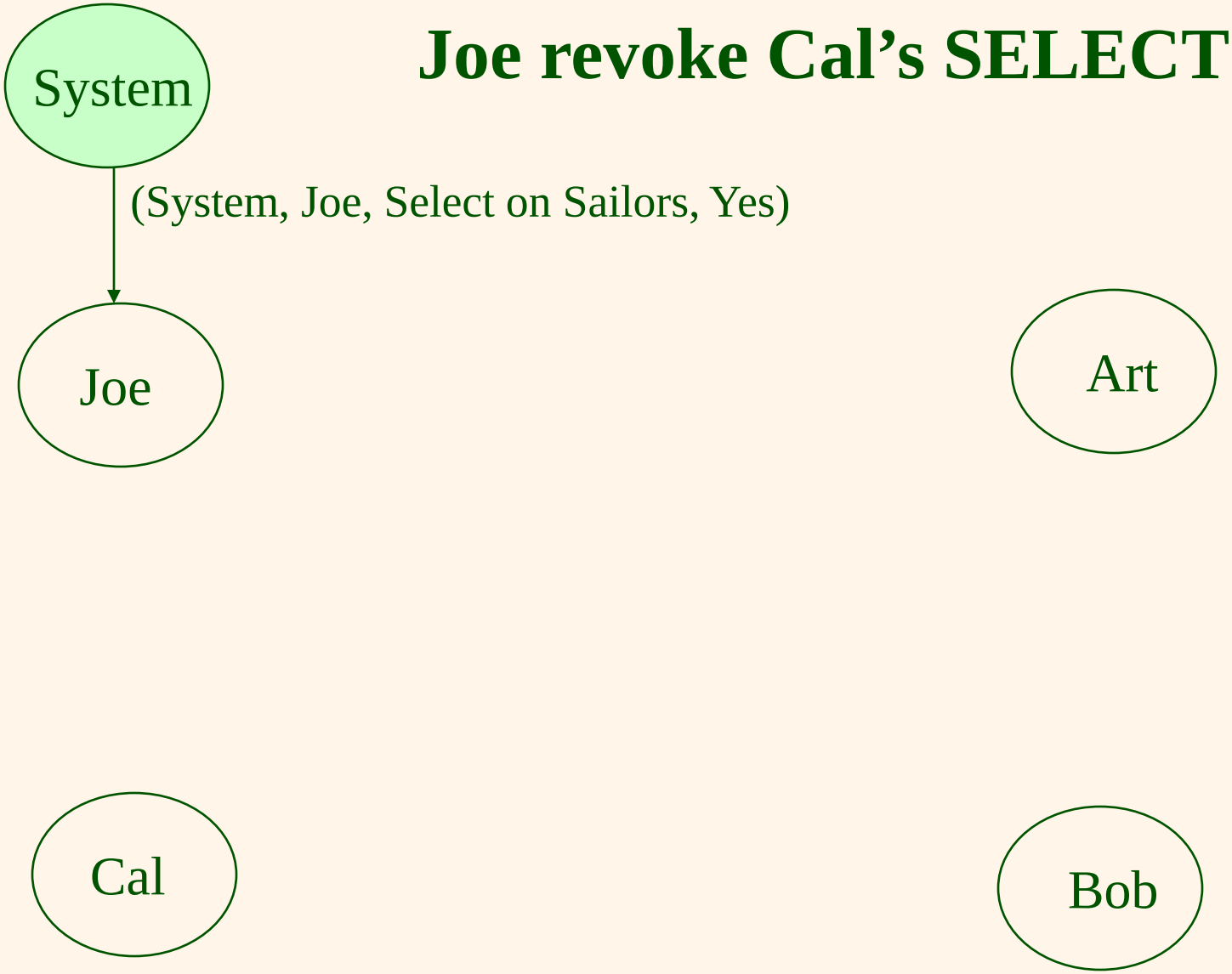
Example Authorization Graph



Authorization Graph after Joe revoke Art's SELECT Privilege



Authorization Graph after Joe revoke Cal's SELECT Privilege



GRANT/REVOKE on Views

- If the creator of a view loses the SELECT privilege on an underlying table, the view is dropped!
- If the creator of a view loses a privilege held with the grant option on an underlying table, (s)he loses the privilege on the view as well; so do users who were granted that privilege on the view!

Role-Based Authorization

- In SQL-92, privileges are assigned to authorization ids, which can denote a single user or a group of users.
- In SQL:1999 (and in many current systems), privileges are assigned to roles.
  - Roles can be granted to users and other roles.
  - Reflects how real organizations work.

Discretionary Access Control

- Susceptible to Trojan Horse schemes
  - Unauthorized user can trick an authorized user to disclose sensitive data
- Example: student Tricky Dan can break into grade table of instructor Trusty Justin.
  - Create a new table called MineAllMine and gives INSERT privilege on this table to Justin
  - Modify code of a DBMS application that Justin often use to do additional things:
    - Read Grades table
    - Write result into MineAllMine

Mandatory Access Control

- Based on system-wide policies that cannot be changed by individual users.
  - Each DB object is assigned a security class.
  - Each subject (user or user program) is assigned a clearance for a security class.
  - Rules based on security classes and clearances govern who can read/write which objects.
- Most commercial systems do not support mandatory access control. Versions of some DBMSs do support it; used for specialized (e.g., military) applications.

Bell-LaPadula Model

- Objects (e.g., tables, views, tuples)
- Subjects (e.g., users, user programs)
- Assign security classes for objects
- Assign clearances for subjects
- Partial order of classes:
  - Top secret (TS), secret (S), confidential (C), unclassified (U)
  - TS > S> C > U

Bell-LaPadula Model

- Simple Security Property
  - Subject S can read object O only if

    class(S) >= class(O)
  - Eg. User with TS clearance can read a table with C clearance, but a user with C clearance is not allowed to read a table with TS classification

Bell-LaPadula Model

- *-Property
  - Subject S can write object O only if

    class(S) <= class(O)
  - Eg. User with S clearance can write only objects with S or TS classification

Bell-LaPadula Model

- Idea is to ensure that information can never flow from a higher to a lower security level.
- E.g., If Dan has security class C, Justin has class S, and the table Branch has class S:
  - Dan's table, Branch, has Dan's clearance, C.
  - Justin's application has his clearance, S.
  - So, the Justin's program cannot write into table Branch.
- MAC rules can be applied in addition to any DAC that are in effect.

Multilevel Relations

- To apply MAC in a relational DBMS, a security class must be assigned to each database object
- Objects can be at the granularity of tables, rows or even individual column values.
- Assume each row is assigned a security class → Multilevel tables
  - Users with different security clearances see a different collection of rows when they access the same table.

Multilevel Relations

bid	bname	color	class
101	Salsa	Red	S
102	Pinto	Brown	C

- Users with S and TS clearance will see both rows
- A user with C clearance will only see the 2nd row
- A user with U clearance will not see any rows.

Multilevel Relations

bid	bname	color	class
101	Salsa	Red	S
102	Pinto	Brown	C

- If key of Boat table is *bid,* and user with clearance C tries to insert <101,Pasta,Blue,C>:
  - Insertion will violate key constraint - two rows with same key 101
  - Disallow insertion will violate security principle – user can infer that there is another object with key 101 that has a class > C! (C cannot see 101 with S)

Multilevel Relations

bid	bname	color	class
101	Salsa	Red	S
102	Pinto	Brown	C
101	Pasta	Blue	C

- Resolve problem by treating class as part of key
  - Users with C clearance will see rows for Pinto and Pasta
  - Users with S or TS clearance will see all 3 rows

Statistical DB Security

- Statistical database
  - Contains information about individuals
  - Allows only aggregate/statistical queries
  - E.g., average age of sailors, rather than Joe's age
- Possible to infer some secret information!
  - If I know Joe is the oldest sailor, I can ask "How many sailors are older than X?" for different values of X until I get the answer 1 → infer Joe's age.
  - Let X=65. If I ask "What is the maximum rating of sailors whose age is greater than 65?" → reveal Joe's rating.

Statistical DB Security

- Resolve violation by requiring that each query must involve at least N rows, for some N ? NO !
  - Repeatedly ask "How many sailors older than X?" until the system rejects the query
  - Identify a set of N sailors, including Joe, that are older than X; let X=55 at this point.
  - Next ask "What is the sum of ages of sailors older than X?" Let result be S1.
  - Next, ask "What is sum of ages of sailors other than Joe who are older than X, plus my age?" Let result be S2.
  - S1 - S2 + my age is Joe's age!

Statistical DB Security

- Ask two queries that involve many of the same sailors.
- Maintain log of all queries, to detect such query pattern.
- Difficult to enforce security in statistical db !