Principles of Distributed Database Systems

M. Tamer Özsu

Patrick Valduriez

Outline

- Distributed Data Control
  - View management
  - Data security
  - Semantic integrity control

Semantic Data Control

- Involves:
  - View management
  - Security control
  - Integrity control
- Objective :
  - Ensure that authorized users perform correct operations on the database, contributing to the maintenance of the database integrity.

Outline

- Distributed Data Control
  - View management
  - Data security
  - Semantic integrity control

View Management

View – virtual relation

- generated from base relation(s) by a query
- not stored as base relations

Example :

```
CREATE VIEW   SYSAN(ENO,ENAME)
AS     SELECT ENO,ENAME
       FROM   EMP
       WHERE  TITLE= "Syst. Anal."
```

EMP

ENO	ENAME	TITLE
E1	J. Doe	Elect. Eng
E2	M. Smith	Syst. Anal.
E3	A. Lee	Mech. Eng.
E4	J. Miller	Programmer
E5	B. Casey	Syst. Anal.
E6	L. Chu	Elect. Eng.
E7	R. Davis	Mech. Eng.
E8	J. Jones	Syst. Anal.

SYSAN

ENO	ENAME
E2	M. Smith
E5	B. Casey
E8	J. Jones

View Management

Views can be manipulated as base relations

Example :

```
SELECT ENAME, PNO, RESP
FROM   SYSAN, ASG
WHERE  SYSAN.ENO = ASG.ENO
```

Query Modification

Queries expressed on views

↓

Queries expressed on base relations

Example :

```
SELECT ENAME, PNO, RESP
FROM   SYSAN, ASG
WHERE  SYSAN.ENO = ASG.ENO
```

↓

```
SELECT ENAME,PNO,RESP
FROM   EMP, ASG
WHERE  EMP.ENO = ASG.ENO
AND    TITLE = "Syst. Anal."
```

ENAME	PNO	RESP
M. Smith	P1	Analyst
M. Smith	P2	Analyst
B. Casey	P3	Manager
J. Jones	P4	Manager

View Updates

- Updatable

```
CREATE VIEW    SYSAN(ENO,ENAME)
AS     SELECT  ENO,ENAME
       FROM    EMP
       WHERE   TITLE="Syst. Anal."
```

- Non-updatable

```
CREATE VIEW    EG(ENAME,RESP)
AS     SELECT  ENAME,RESP
       FROM    EMP, ASG
       WHERE   EMP.ENO=ASG.ENO
```

View Management in Distributed DBMS

- Views might be derived from fragments.
- View definition storage should be treated as database storage
- Query modification results in a distributed query
- View evaluations might be costly if base relations are distributed

Materialized View

- Origin: snapshot in the 1980's
  - Static copy of the view, avoid view derivation for each query
  - But periodic recomputing of the view may be expensive
- Actual version of a view
  - Stored as a database relation, possibly with indices
- Used much in practice
  - DDBMS: No need to access remote, base relations
  - Data warehouse: to speed up OLAP
    - Use aggregate (SUM, COUNT, etc.) and GROUP BY

Materialized View Maintenance

- Process of updating (refreshing) the view to reflect changes to base data
  - Resembles data replication but there are differences
    - View expressions typically more complex
    - Replication configurations more general
- View maintenance policy to specify:
  - When to refresh
  - How to refresh

When to Refresh a View

- Immediate mode
  - As part of the updating transaction, e.g. through 2PC
  - View always consistent with base data and fast queries
  - But increased transaction time to update base data
- Deferred mode (preferred in practice)
  - Through separate refresh transactions
    - No penalty on the updating transactions
  - Triggered at different times with different trade-offs
    - Lazily: just before evaluating a query on the view
    - Periodically: every hour, every day, etc.
    - Forcedly: after a number of predefined updates

Outline

- Distributed Data Control
  - View management
  - Data security
  - Semantic integrity control

Data Security

- Data protection
  - Prevents the physical content of data to be understood by unauthorized users
  - Uses encryption/decryption techniques (Public key)
- Access control
  - Only authorized users perform operations they are allowed to on database objects
  - Discretionary access control (DAC)
    - Long been provided by DBMS with authorization rules
  - Multilevel access control (MAC)
    - Increases security with security levels

Discretionary Access Control

- Main actors
  - Subjects (users, groups of users) who execute operations
  - Operations (in queries or application programs)
  - Objects, on which operations are performed
- Checking whether a subject may perform an op. on an object
  - Authorization= (subject, op. type, object def.)
  - Defined using GRANT OR REVOKE
  - Centralized: one single user class (admin.) may grant or revoke
  - Decentralized, with op. type GRANT
    - More flexible but recursive revoking process which needs the hierarchy of grants

Problem with DAC

- A malicious user can access unauthorized data through an authorized user
- Example
  - User A has authorized access to *R* and *S*
  - User B has authorized access to *S* only
  - B somehow manages to modify an application program used by A so it writes *R* data in *S*
  - Then B can read unauthorized data (in *S*) without violating authorization rules
- Solution: multilevel security based on the famous Bell and Lapuda model for OS security

Multilevel Access Control

- Different security levels (*clearances*)
  - *Top Secret* > *Secret* > *Confidential* > *Unclassified*
- Access controlled by 2 rules:
  - No read up
    - subject *S* is allowed to read an object of level *L* only if *level(S)* ≥ *L*
    - Protect data from unauthorized disclosure, e.g. a subject with secret clearance cannot read top secret data
  - No write down:
    - subject *S* is allowed to write an object of level *L* only if *level(S)* ≤ *L*
    - Protect data from unauthorized change, e.g. a subject with top secret clearance can only write top secret data but not secret data (which could then contain top secret data)

MAC in Relational DB

- A relation can be classified at different levels:
  - Relation: all tuples have the same clearance
  - Tuple: every tuple has a clearance
  - Attribute: every attribute has a clearance
- A classified relation is thus multilevel
  - Appears differently (with different data) to subjects with different clearances

Example

PROJ*: classified at attribute level

PNO	SL1	PNAME	SL2	BUDGET	SL3	LOC	SL4
P1	C	Instrumentation	C	150000	C	Montreal	C
P2	C	DB Develop.	C	135000	S	New York	S
P3	S	CAD/CAM	S	250000	S	New York	S

PROJ* as seen by a subject with confidential clearance

PNO	SL1	PNAME	SL2	BUDGET	SL3	LOC	SL4
P1	C	Instrumentation	C	150000	C	Montreal	C
P2	C	DB Develop.	C	Null	C	Null	C

Distributed Access Control

- Additional problems in a distributed environment
  - Remote user authentication
    - Typically using a directory service
      - Should be replicated at some sites for availability
  - Management of DAC rules
    - Problem if users' group can span multiple sites
      - Rules stored at some directory based on user groups location
      - Accessing rules may incur remote queries
  - Covert channels in MAC

Covert Channels

- Indirect means to access unauthorized data
- Example
  - Consider a simple DDB with 2 sites: C (confidential) and S (secret)
  - Following the “no write down” rule, an update from a subject with secret clearance can only be sent to S
  - Following the “no read up” rule, a read query from the same subject can be sent to both C and S
  - But the query may contain secret information (e.g. in a select predicate), so is a potential covert channel
- Solution: replicate part of the DB
  - So that a site at security level *L* contains all data that a subject at level *L* can access (e.g. S above would replicate the confidential data so it can entirely process secret queries)

