Integrity and Security

Jan Chomicki
University at Buffalo

Outline

Transactions

Transaction

Execution of a user program in a DBMS.

Transaction properties

- **A**tomicity: all-or-nothing execution
- **C**onsistency: database consistency is preserved
- **I**solation: concurrently executing transactions have no effect on one another
- **D**urability: results survive failures.

Transaction outcome

- COMMIT: success, effects made permanent
- ROLLBACK: failure, effects removed.

Integrity constraints and triggers

Maintaining the logical integrity of the database.

Integrity constraints

- logical conditions that database instances must satisfy
- maintained by the DBMS

Triggers

- rules for enforcing integrity
- executed by the DBMS
- flexible reaction to integrity violations

Integrity constraints

Column and table constraints

- CHECK constraints
- key constraints
- foreign keys
- associated with a table but can refer to multiple tables
- violated only by insertion/update to the same table.

Assertions

- maintained across multiple tables

Checking mode

- immediate: after an operation
- deferred: after transaction ends

Referential integrity actions (SQL:1999)

Modifications violating a foreign key constraint

- referencing table: *disallowed*
- referenced table:
  - events:
    - `ON UPDATE`
    - `ON DELETE`
  - actions:
    - `SET DEFAULT`
    - `SET NULL`
    - `CASCADE`
    - `NO ACTION` (default: change not made if constraint ultimately violated)
    - `RESTRICT` (no temporary violations)

Active databases

Database become active when augmented with active rules (triggers).

Basic format (ECA rules)

`on` *Event* `if` *Condition* `then` *Action*

Compare with

- integrity constraints
- referential integrity actions

Trigger execution

Execution cycle

```
while there are triggered rules do
    find a triggered rule R
    evaluate the condition of R
    if the condition is true
       then execute the action of R
```

Execution granularity

- *smallest* database operation
- data manipulation command
- at the end of a transaction

Triggers in SQL:1999

Defined using `CREATE TRIGGER`, associated with tables.

Triggering operation

- `INSERT`, `DELETE`, `UPDATE`
- execution mode: `BEFORE` or `AFTER` the triggering statement.

Condition

- arbitrary SQL predicate
- can reference new/old versions of affected rows or tables.

Granularity

- row-level (executed once for each modified row)
- statement-level (executed once for each statement)

Action

- one or more SQL statements

Execution

Trigger execution order determined by their *definition order.*

`BEFORE` triggers

- executed immediately
- cannot modify the database.

`AFTER` triggers

- queued
- fire after integrity checks and the execution of referential integrity actions.

Sequencing

1. execution of `BEFORE` triggers
2. execution of transaction
3. execution of referential integrity actions
4. constraint evaluation
5. execution of `AFTER` triggers

Views

Updatable views in SQL

- a single `SELECT` from some relation R
- R cannot appear in subqueries
- SELECT list has to contain enough attributes that every tuple inserted into the view can be filled with nulls or default values (this implies that the list contains the primary key)

View maintenance

- `INSTEAD` triggers

Authorization in SQL:1999

Privileges for accessing/modifying data

- reading data from a relation/view
- inserting/updating/deleting data in a relation/view
- creating/dropping relations
- creating/dropping views
- adding/dropping columns
- referencing a relation (foreign keys)
- roles

Granting and revoking privileges

Grant

`grant` *privilege list* `on` *relation or view name*
`to` *user/role list* [`with grant option`]

Revoke

`revoke` *privilege list* `on` *relation or view name*
`from` *user/role list* [`restrict` | `cascade`]

Checking authorization

Authorization graph

- nodes: users + privileges
- edges: authorizations granted (and not revoked)

U has authorization to do A iff there is a path authorizing A from the node that has A because of the database element in question to U.

Authorization on views

To compute view contents

- read privileges on the underlying relations

To modify a view

- appropriate modification privileges on the underlying relations