CSC 355 Database Systems
Lecture 14

Eric J. Schwabe

School of Computing, DePaul University

Spring 2020

Today:

- Last Relational Database Design Example
- Introduction to Triggers

Relational Database Design

- Starting with $R(A_1, A_2, \ldots, A_n)$, the decomposition $D = \{R_1, R_2, \ldots, R_m\}$ should satisfy the following conditions:
  1. The union of the R_i's is R
  2. Each of the R_i's is in BCNF (or 3NF)
  3. D has the dependency preservation property
  4. D has the lossless join property

Comparison of Algorithms

- BCNF decomposition algorithm:
  - No redundancy in relations
  - Dependency preservation not guaranteed
  - Lossless join guaranteed
- 3NF decomposition algorithm:
  - Some redundancy may remain in relations that have multiple candidate keys
  - Dependency preservation guaranteed
  - Lossless join guaranteed

Final Decomposition Example

- INVOICE (OrderID, OrderDate, CustomerID, Name, Address, ProductID, Description, Material, Price, Quantity)
  - Review functional dependencies, candidate keys
  - Is INVOICE in BCNF? Is it in 3NF?
  - Construct BCNF and 3NF decompositions
  - Verify dependency preservation and lossless join

Relational Model

- As a data model, the relational model must describe three things about stored data:
  - Structure of the data: A collection of linked two-dimensional tables
  - Operations on the data: Described by SQL statements
  - Constraints on the data: Domain, key, entity integrity, referential integrity, check … what else?

Assertions

- An assertion is a condition that cannot be false for any state of the database
  - If deferred, just at the end of each transaction…
- Can involve any tables in the database

```
CREATE ASSERTION SalaryCap CHECK
( 1000000 >=
        (SELECT SUM(Salary) FROM WORKER ) );
```

- Assertions are not supported by major DBMSs

Triggers

- A trigger is a procedure that is executed in response to a particular database operation

```
CREATE TRIGGER SalaryCap AFTER INSERT ON WORKER
BEGIN
        SELECT SUM(Salary) INTO total FROM WORKER;
        IF (total > 1000000) THEN
                ERROR('Million Dollar Limit Exceeded');
        END IF;
END;
```

Triggers

- Triggers allow general responses to changes in the database state, e.g.:
  - Enforcement of business rules
  - Notification of events
  - Maintenance of derived information
  - Maintenance of replicated data
  - Implementation of multi-table constraints

Triggers

- A trigger definition must specify:
  - An *event* (e.g., insert, delete, update) that causes the trigger to fire
  - A *condition* that is tested (on old and/or new state) to decide whether or not the trigger will respond
  - An *action* (PL/SQL block or stored procedure) that may be executed in response

Oracle Trigger Syntax

CREATE [OR REPLACE] TRIGGER *Name*
BEFORE/AFTER
    INSERT OR DELETE OR UPDATE [OF *Attribute*] ON *TABLE*
[REFERENCING
    OLD AS *OldName*
    NEW AS *NewName*]
[FOR EACH ROW]
[WHEN (*condition*)]
DECLARE
    …variable declarations…
BEGIN
    …PL/SQL statements…
END;
/

Oracle Trigger Syntax

- BEFORE
  - Indicates that queries on *TABLE* will be done on the state of the table before the triggering operation executes
- AFTER
  - Indicates that queries on *TABLE* will be done on the state that the table would be in after the triggering operation executes

Oracle Trigger Syntax

- INSERT OR DELETE OR
  UPDATE [OF *Attribute*] ON *TABLE*
  - What operation(s) will cause the trigger to fire?
  - Trigger will fire in response to any INSERT or DELETE
  - Trigger may be set to fire in response to any UPDATE, or only an UPDATE of a particular attribute

Oracle Trigger Syntax

- FOR EACH ROW
  - If not included, indicates a *statement-level* trigger that will fire just once for the entire operation
  - If included, indicates a *row-level* trigger that will fire once for each row that is modified
    - … so an UPDATE or DELETE that applies to multiple rows will cause the trigger to fire more than once for the operation …

Oracle Trigger Syntax

- REFERENCING OLD AS *OldName*, NEW AS *NewName*
  - The original and modified states of the row being operated upon are called “old” and “new” unless you change them (used for row-level triggers only)
    - INSERT has only “new”, but no “old”
    - DELETE has only “old”, but no “new”
    - UPDATE has both “old” and “new”

Oracle Trigger Syntax

- WHEN (*condition*)
  - Condition tested to see whether or not the trigger action will actually execute
    - Statement-level: can query original or modified table state depending on whether BEFORE or AFTER is used
    - Row-level: can reference original and modified row states with “old” and “new” (INSERT has only “new”, DELETE has only “old”, UPDATE has both!)

Oracle Trigger Syntax

- *PL/SQL statements*
  - This block of code is executed when the trigger fires and the WHEN condition is satisfied
  - It may include:
    - SQL statements
    - PL/SQL statements
    - Calls to built-in or user-defined procedures/functions

Oracle Trigger Restrictions

- new and old can only refer to row states, and can only be used for row-level triggers
  - Use new and old in WHEN condition, but :new and :old elsewhere
- PL/SQL statements in a row-level trigger cannot query or modify the table that triggered the action
- Subqueries are not allowed in WHEN

Trigger Examples

- Salary Cap:
  - Trigger cancels any operation that causes the company's total budget for salaries to exceed $1,000,000 (statement-level)
- Departmental Budgets:
  - Trigger maintains current totals of the salaries of all employees in each department (row-level)

Next:

- Database Programming in PL/SQL
- Back to Triggers