DATABASE REDESIGN

revised by 김태연

KROENKE AND AUER - DATABASE PROCESSING, 13th Edition © 2014 Pearson Education, Inc.

OBJECTIVES

- To understand the need for database redesign
- To understand reverse engineering
- To be able to change table names & columns
- To be able to change relationship cardinalities
- To be able to change relationship properties
- To be able to add and delete relationships

NEED FOR DATABASE REDESIGN

- Database redesign is necessary:
  - To fix mistakes made during the initial database design.
  - To adapt the database to changes in system requirements.
- Because information systems and organizations create each other, a new information system will cause changes in systems requirements:
  - When a new system is installed, users can behave in new ways.
  - As the users behave in the new ways, they will want changes to the system to accommodate their new behaviors.

DATABASE REDESIGN

- Three principles for database redesign:
  - Measure twice and cut once: understand the current structure and contents of the database before making any structure changes.
  - Test the new changes on a test database before making real changes.
  - Create a complete backup of the operational database before making any structure changes.
- Technique:
  - Reverse Engineering (RE)
  - Refactoring Database - rename

REVERSE ENGINEERING (RE)

- Reverse engineering (RE) is the process of reading and producing a data model from a database schema.
- A reverse engineered (RE) data model:
  - Provides a basis to begin the database redesign project.
  - Is neither truly a conceptual nor an internal schema as it has characteristics of both.
  - Should be carefully reviewed because it almost always has missing information.

DATABASE BACKUP AND TEST DATABASES

- Before making any changes to an operational database:
  - A complete backup of the operational database should be made.
  - Any proposed changes should be thoroughly tested.
- Three different copies of the database schema used in the redesign process:
  - A small test database for initial testing
  - A large test database for secondary testing
  - The operational database

DATABASE REDESIGN CHANGES

- Changing tables and columns
  - Changing table names
  - Adding and dropping table columns
  - Changing data type or constraints
  - Adding and dropping constraints
- Changing relationships
  - Changing cardinalities
  - Adding and deleting relationships
  - Adding and removing relationships for denormalization

CHANGING TABLES AND COLUMNS

- Although SQL or DBMS specific commands exist, there is no good command to change a table name except in the most simple cases.
  - The table needs to be re-created under the new name, tested, and the old table is dropped.
- Changing a table name has a surprising number of potential consequences.
  - Therefore, using views defined as table aliases is more appropriate.
  - Only views that define the aliases would need to be changed when the source table name is changed.

CHANGING MINIMUM CARDINALITIES

- On the parent side:
  - To change from zero to one, change the foreign key constraint from NULL to NOT NULL.
  - Can only be done if all the rows in the table have a value.
- To change from one to zero, change the foreign key constraint from NOT NULL to NULL.
- On the child side:
- Add (to change from zero to one) or drop (to change from one to zero) triggers that enforce the constraint.

CHANGING MAXIMUM CARDINALITIES

- 1:1 to 1:N
  - If the foreign key is in the correct table, remove the unique constraint on the foreign key column.
  - If the foreign key is in the wrong table, move the foreign key to the correct table and do not place a unique constraint on that table.
- 1:N to N:M
  - Build a new intersection table and move the key and foreign key values to the intersection table

FORWARD ENGINEERING

- Forward engineering is the process of applying data model changes to an existing database.
- Results of forward engineering should be tested before using it on an operational database.
- Some tools will show the SQL that will execute during the forward engineering process:
  - If so, that SQL should be carefully reviewed.