A Database Design and Implementation Project Using Design Patterns

A typical class project in a database management course usually involves the following four major steps (sub-projects):

- Analyze the database problem and capture the requirements specification in a conceptual model such as Entity-Relationship Model;
- Map the conceptual schema to a logical database model (e.g., relational schema) and make use of various database design concepts (e.g., normalization);
- Create and populate relational tables;
- Create stored queries and ad hoc queries to solve various database problems.

In transferring knowledge between databases and software engineering, the first knowledge to transfer is the design and implementation processes. For example, the Unified Process [ref] for object-oriented system can be used to illustrate the design and implementation concepts in building a database system from requirements, specification, design, to implementation/code generation. Pedagogically, the software engineering life cycle should be introduced in database management course.

We can follow the above four processes in the design and implementation of a database system but introduce software engineering and design pattern concepts in each process.

Problem analysis and the construction of Entity-Relationship Model: Software engineering and design pattern introduced in this process include:

- UML model for ER model – entities as classes, table entries (tuples) as objects, relationships as associations, primary and foreign keys as object identifiers, etc.
- Entities and relationships in ER model as examples of Abstract Factory.
- Aggregation/composition in EER model as examples of Composite.

Conceptual schema and logical database model mapping: Software engineering and design pattern introduced in this process include:

- UML model – mapping between class diagram and class constructs in C++ or Java can be used to illustrate the mapping between classes and table schema.
- Relational table creation as an example of Builder.
- Decoupling of logical database design (tables) from physical database design (array, linked-list, vectors) as an example of Bridge.

Creating and populating relational tables: Software engineering and design pattern introduced in this process include:

- UML model – object diagrams for object creation.
- OCL [ref] for integrity constraints.
- Views as examples of Façade and Observer.
- Object creation as an example of Prototype.

Problem solving queries: Software engineering and design pattern introduced in this process include:
• UML model – Use Case Models, Narratives and Scenarios, Interaction Diagrams (Sequence and collaboration diagrams) for users’ requests.
• Queries involve the interactions of objects and thus apply to any of the Behavioral Patterns.