Conceptual, Logical, and Physical design of Persistent Data using UML

The database needs a structure definition to be able to store data and to recognize the content and be able to retrieve information. The structure has to be developed for the need of applications, which help us perform a business process to achieve an added value for the customer.

Database implementation is the final step in developing a business supporting application. It has to comply with the requirements of the business process, which is the first abstraction of the view of the database.

Conceptual database design

Several definitions describe the conceptual design more or less precise. The one I like most is:

The process of constructing a model of the information used in an enterprise, independent of all physical considerations. [Conn1998]

But still, this definition is now strong enough. It does not express the real value of conceptual modeling - the business process relation. I like the following definition better.

Conceptual database design is the process of building a model of the essential part of the enterprise business process and the used information, independent of all physical considerations.

The focus of the conceptual database design has to answer the question of the reason for persistent data. The knowledge of the reason allows the data analysts and data modeler to model the right database right. Choosing the right database involves the decision on the kind of data storage to store information, building the database right concerns the internal structure of the data storage.

The business model contains the business process and the business artifacts. A business use case represents a specific workflow in the system. The enterprise business process is the summary of all of the business use cases.

Business use cases are developed in the use case realization, which is a textual and visual description of how the business use case is realized in terms of collaborating objects. Its main purpose is to summarize the diagrams connected to the business use case, and to explain how they are related.

The business use case evolves into activity diagrams, which explore the ordering of tasks or activities that accomplish business goals and that satisfy commitments between external business actors and internal business workers. An activity may be a manual or automated task that completes a unit of work.

The activity diagram provides the best way of understanding the information sources for every workflow. It describes where the information comes from, and how it relates to the workflow.

The collaboration diagram provides a view of correlation between objects participating in an interaction. It can show the message they send to each other, which specifies the kind of the relation. Persistent objects, most commonly marked with the stereotype <>, are able to store data, which has to be specified in the documentation of the object. This artifact is important for the database design.

The complete sequence of the messages between objects in one scenario in a time sequence is specified in a sequence diagram. A scenario is one flow through the use case. The message itself can carry data in either direction as parameters or result of the invocation. The message and the carried data define the information you need and the requested public access to the data. The frequency of the implemented use case and the repetition of the message define the frequency of the access.

The conceptual database design handles the data view of the business process. It is a part of the business model of the entire enterprise. It defines the global need for persistent data, but not the physical details of the implementation. It includes use cases and use case realization, including activity, collaboration, and sequence diagrams, but does not require all of the content.

Logical database design

The process of constructing a model of the information used in an enterprise based on a specific data model, but independent of a particular DBMS and other physical considerations. [Conn1998]

The decision on the kind of data storage is the step from the conceptual to logical database design, as the logical database design describes a DBMS related representation, and the conceptual design the target independent process including the logic of the application.

This decision is not always easy when talking about long-term transient data, which is used just until the application stops and not used very often. Examples for long-term transient data are web-server component data, like transaction data, user data, etc.

I would like to expand the definition and include the granularity and language of description. With the logical database design the natural language and granularity of objects has to be used. Any differentiation from the base element would influence the result of the design and focus already on a specific architecture.

The process of constructing a model of information used in an enterprise based on a specific data model, using natural objects of information and natural associations between them. The model of information is independent of a particular implementation and other physical consideration.

The logical database design is the model of valuable information. It describes the information itself and the relationships between information. The kind of relationships is not limited, because any limitation would affect the structure of information.

The description of the information is implementation neutral. It uses abstraction to depersonalize objects from the conceptual database design. Objects of the same kind are grouped together to one persistent class. The objects are still at the same granularity level as the global requirements.

Persistent classes contain attributes, which represent the basic element of persistent storage. An attribute can store any kind of data independent of the later physical limitation. DBMS independent attribute types are used on the logical level to prevent any physical influence.

The accessibility of the persistent data is defined by operations on a class. In the design process operations evolve from messages at interaction diagrams. Operations can be public or private, in which case they deal with the internal integrity of the information. Private operations are mostly defined out of the complexity of the business process, which is best recognized in the use case realization. They are rules for complex relations between information.

The association between classes can represent any kind of influence. The association does not have any influence on the attributes, and the attributes do not have any influence on the association.

Since the logical model does not consider any particular implementation, the associations may include aggregation (often called part of association), inheritance (type of association), and any kind of cardinality (including m to n associations).

The logical design is most commonly expressed in a class diagram, which is a partial view on the whole design. The resulting artifacts of the logical design are persistent classes and the associations between classes, as well as the essential constraints on the data.

The logical database design handles the information view on the enterprise. It contains most important information in the enterprise as well as the associations between information. As a difference to the conceptual database design, logical database design does not contain the process itself, because the process is further evolved in the application design.

Physical database design

The process of producing a description of the implementation of the database on secondary storage. It describes the storage structures and access methods used to achieve efficient access to the data. [Conn1998]

The physical database design defines the way to implement the data storage. It is target dependent and complies with the storage technology, the instance of the database, the architecture, and the implementation rules.

The physical database design has to remodel the artifacts of the logical design according to the used storage technology, which can be hierarchical, network, relational, object-relational, object-oriented, or multi-dimensional. And I am sure more will exist in future. The challenge is to design conversion rules between the independent logical model and technology dependent physical model.

The aim of the physical design is for the relational database to define the tables and detailed constraints for the in the logical database design specified data model. The columns of a table are specified with the details on internal representation.

Associations are translated to relationships, which are represented by key constraints. Operations can be implemented as all kinds of access methods. Together they form information out of pure data.

An important view on data is the security, which can be easily defined out of the use case realizations and the logical database artifacts. It is important to have the global overview of the application and data storage to be able to design security schemas.

The resulting artifact of the physical database design is the exact description of the implementation.

Summarizing the Workflows

The database design workflows have to be separated because of a different focus on modeling. They require different skills.

The critical moment on the design of the database is the interface between the workflows. The database design process is creative work, supported by great tools, which help to keep the design artifacts in a consistent state.

The result of any of the database design workflows is an input to the next workflow. Every workflow gets the artifacts from the previous one, and from other sources, like the stakeholder, the experience, and general patterns.

Going from conceptual database design to the logical database design requires lots of individual decisions. The designer has to decide about the grouping of objects, the quantity and quality of information, and the relations between information. He uses lots of patterns to develop the logical design. This process can be tracked with a tool, which is able to reconstruct the evolution of the logical design.

Physical model is developed out of the logical model using conversion rules. It is the physical incarnation of logical design. The rules have to convert the technology independent logical design into target dependent physical design considering all of the constraints of the target.

Rational Unified Process® suggests a full set of rules to convert from the conceptual database design to the logical database design. According to the need of the business process this rules as well as the development process can be customized on the specific needs of the implementation. Rational has implemented a full set of rules to convert from the logical design to the physical design and vice versa. It includes rules to convert classes to tables, attributes to columns, and associations to relationships, constraints, and tables.

UML offers new and better understanding of the enterprise and the complete semantics for the database design. It is a new technology to design the complete application including the database without any restrictions compared to existing ER semantics. It makes the design more accurate and the communication easier.

References

[Conn1998] Database Systems - A Practical Approach to Design, Implementation, and Management; Thomas Connolly, Carolyn Begg, Anne Strachan; Addison-Wesley; ISBN 0-201-34287-1