A Crash Course in Metaphysics for the Database Designer

John M. Artz
George Washington University

Information modeling is the first stage of database design. The central problem in information modeling is the identification of the proper set of objects to be included in the model. The wrong set of objects can lead to data integrity problems and incorrect information being delivered to unwitting users. A subtle but important point is that the information model represents object classes which are groupings of objects with similar characteristics. At the heart of the problem of identifying object classes is the question "Do object classes exist in the world waiting to be discovered by the information modeler, or are they constructed by the modeler based on abstractions drawn from the application domain?" Although most modeling approaches implicitly assume that object classes do exist, an analysis of the opinions of several major western philosophers from Plato to Wittgenstein suggest that this assumption may be flawed. If this is the case, the practice of information modeling changes dramatically.

The ancient Greek philosopher Aristotle was also a teacher at an academy he founded in Greece called the Lyceum. Aristotle would lecture in the morning on the natural sciences which he called 'physics'. In the afternoon he would lecture on ethics, politics, and aesthetics which he called metaphysics, meaning after physics. So, in some sense, metaphysics just means the afternoon lectures. However, the term has actually come to refer to that speculative branch of philosophy that attempts to get at the underlying nature of reality. Metaphysics is concerned with such arcane problems as the nature of the universe, causality, the concept of identity, and the mind/body problem. So what does all this have to do with database design? There is another concern of metaphysics called the problem of universals which is also the central problem in database design. While most designers are completely unaware of the fact that they are grappling with the problem of universals, they still face it everyday in routine design decisions. Thus, some understanding of this esoteric problem will help database designers design better databases. Consequently, the purpose of this paper is to provide a crash course in metaphysics for database designers and show how the problem of universals is central to good database design.

Information Modeling

Information modeling is the first stage of database design, sometimes referred to as conceptual database design. At this stage, the database designer attempts to identify the "things" of interest to database users and the relevant relationships between them. The "things" of interest can be called entities [Chen] [Teorey] [Nijssen and Halpin] or objects [Flavin] [Shlaer and Mellor] [Veryard, 1984 & 1992] depending on the modeling methodology. Object is the term that will be used in this paper because it is more general and more recent. The database designer constructs an information model by interviewing potential database users about their information needs and the resulting model is a record of the objects of interest in the application domain, important attributes of those objects, and the significant named relationships between those objects. For example, in a university domain, objects of interest include: Students, Professors, and Courses. Important attributes of Students are their Student_Id, Name, and Major_Field, among others. Named relationships between objects are Students Take Courses, and Professors Teach Courses.
The identification of the objects in the model is actually a little trickier than it initially appears. As an information modeler examines the university domain, he or she notices individuals - Bob, Jane, Lee - who take classes. These individuals also have some common attributes such as student number, name, and major. So the modeler groups these individuals into a class called **Student**, and assigns the common attributes to the class. The information model actually models object classes rather than objects although few modelers make this distinction. It is the object class that is represented in the database via tables, whereas individual objects are represented by rows.

There are several reasons why the distinction between an object and an object class is often overlooked in information modeling. First, it is frequently the case that a new database system is replacing an older system of some kind in which the classes (often record types) are already defined. Second, we frequently use the object type to refer to individual objects so the distinction is lost in usage. For example, we refer to a student as a student rather than a person who is a student. The first reference allows only one grouping - Student. The second reference allows other grouping such as: a person who is a customer of the university, or a person who enrolls in classes. Finally, certain classes seem so natural that it is hard to see what other way the individuals in that class could be identified. These classes are normally classes with which we have first hand physical experience [Rosch].

The problem that occurs in going from objects to object classes is that individual objects can be grouped in many different ways. The people taking classes could have been grouped in an object class called People, or Freshman, or Paying Customers of the University. Each of these alternative groupings would have included possibly different sets of individual objects. There would have been different class attributes which would have provided different kinds of information to the database users. Having recognized that several potential groupings are possible we have to ask the question - How do we form these groupings and how do we know that a particular grouping is correct? Is there a single correct way to group objects in a domain or are their multiple ways? If there are multiple ways how do we know which one is the best? From a practitioner’s perspective this issue can be described as semantic absolutism versus semantic relativism.

“There are two philosophical attitudes toward data modelling known respectively as semantic relativism and semantic absolutism. According to the absolutist way of thinking, there is only one correct or ideal way of modelling anything: each object in the real world must be represented by a particular construct. Semantic relativists, on the other hand, believe that most things in the real world can be modelled in many different ways, using any of the basic constructs.” [Veryard, 1984, pg. 7]

This distinction has also been characterized as an objectivist view versus a subjectivist view [Klien and Hirschheim]. Yet the underlying problem is the *problem of universals* which has been a central problem in metaphysics since the time of Plato.

It is important that the information modeler identify the correct set of object classes in a domain because a failure to do so could result in database integrity problems or the delivery of bad information to unwary users. For example, consider the university domain and the object class **Student**. On the face of it, it seems fairly straightforward that a student is an important object in the university database. But what is a **Student**? How does a person become a **Student**? How does a person leave the class of **Students**? Is a person who is enrolled on a part-time basis still a **Student**? Is a person who has not enrolled for the current semester still a **Student**? Is a person who sits in classes and learns things, but has not paid, still a **Student**? Is a person who pays, but does not attend classes still a **Student**? And so on.

One of the problems that may occur is that the domain maybe modeled incorrectly and that a user of the database may receive incorrect information in response to an apparently valid query. For example, if a user of the university database inquires about the number of students in the university, he or she will receive a number in response. What that number means will depend on how the information is represented in the database, more specifically how the object class **Student** was defined.

Following along with the university example, suppose the conceptual designer decides to have the object classes: **Student** and **Research Assistant**. If someone is both a Student and a Research Assistant should they belong to one of the two classes or both? If they are represented in both classes, then there is duplicate information in the database which then allows the possibility that the database may become inconsistent. If this person moves, for example, and the **Research Assistant** class is updated, then any mailings generated from the **Student** class will go to the old address. Procedures could be put in place so that the **Student** information is automatically updated, but this solution would lead to complexity in the database. Research Assistants may also be Teaching Assistants. Teaching Assistants may also be Students, and Professors may also be Students. Procedures would have to be established for all these possibilities. Hence duplicating the information is undesirable. If the person who is both a Student and a Research Assistant is represented only in the **Student** class, then incorrect information will be delivered when someone queries the **Research Assistant** class to determine how many Research Assistants the university employs. So the identification of correct object classes is important to the integrity of the database and the integrity of the information it provides.

The problem of grouping individual objects into object classes goes well beyond information modeling. It encompasses a deep philosophical issue that divides philosophers...
Related Content

INDUSTRY AND PRACTICE: Another Look at On Time and Within Budget: An Agency Theory Explanation
www.igi-global.com/article/industry-practice-another-look-time/51221?camid=4v1a

TOS: A Temporal Object-Oriented System
www.igi-global.com/article/tos-temporal-object-oriented-system/51138?camid=4v1a

Incremental Discovery of Fuzzy Functional Dependencies
www.igi-global.com/chapter/incremental-discovery-fuzzy-functional-dependencies/20370?camid=4v1a

Privacy-Preserving Outsourced Similarity Search
www.igi-global.com/article/privacy-preserving-outsourced-similarity-search/118088?camid=4v1a