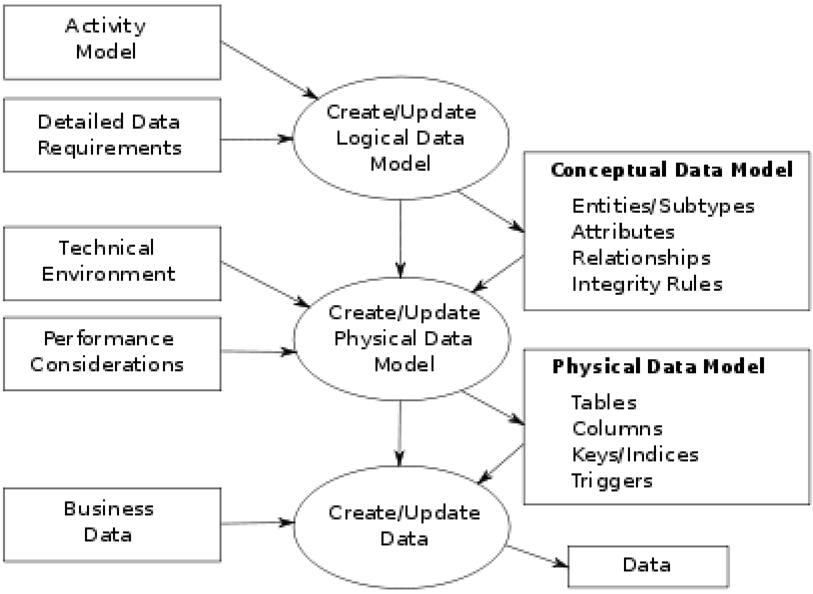
Data modeling

Data modeling in software engineering is the process of creating a data model for an information system by applying certain formal techniques.



The data modeling process. The figure illustrates the way data models are developed and used today . A conceptual data model is developed based on the data requirements for the application that is being developed, perhaps in the context of an activity model. The data model will normally consist of entity types, attributes, relationships, integrity rules, and the definitions of those objects. This is then used as the start point for interface or database design.[1]

Overview

Data modeling is a process used to define and analyze data requirements needed to support the business processes within the scope of corresponding information systems in organizations. Therefore, the process of data modeling involves professional data modelers working closely with business stakeholders, as well as potential users of the information system.

There are three different types of data models produced while progressing from requirements to the actual database to be used for the information system.[2] The data requirements are initially recorded as a conceptual data model which is essentially a set of technology independent specifications about the data and is used to discuss initial requirements with the business stakeholders. The conceptual model is then translated into a logical data model, which documents structures of the data that can be implemented in databases. Implementation of one conceptual data model may require multiple logical data models. The last step in data modeling is transforming the logical data model to a physical data model that organizes the data into tables, and accounts for access, performance and storage details. Data modeling defines not just data elements, but also their structures and the relationships between them.[3]

Data modeling techniques and methodologies are used to model data in a standard, consistent, predictable manner in order to manage it as a resource. The use of data modeling standards is strongly recommended for all projects requiring a standard means of defining and analyzing data within an organization, e.g., using data modeling:

- to assist business analysts, programmers, testers, manual writers, IT package selectors, engineers, managers, related organizations and clients to understand and use an agreed upon semi-formal model that encompasses the concepts of the organization and how they relate to one another
- to manage data as a resource
- to integrate information systems
- to design databases/data warehouses (aka data repositories)

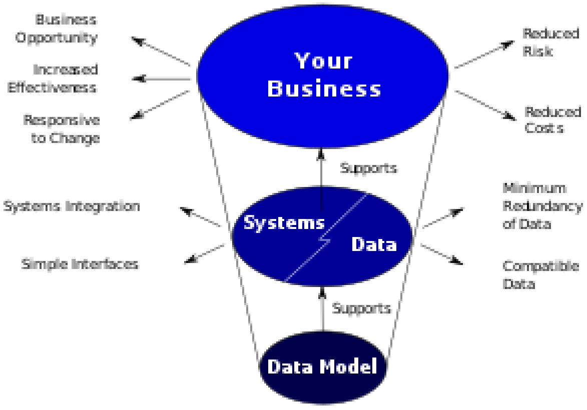
Data modeling may be performed during various types of projects and in multiple phases of projects. Data models are progressive; there is no such thing as the final data model for a business or application. Instead a data model should be considered a living document that will change in response to a changing business. The data models should ideally be stored in a repository so that they can be retrieved, expanded, and edited over time. Whitten et al. (2004) determined two types of data modeling:[4]

- Strategic data modeling: This is part of the creation of an information systems strategy, which defines an overall vision and architecture for information systems. Information technology engineering is a methodology that embraces this approach.
- Data modeling during systems analysis: In systems analysis logical data models are created as part of the development of new databases.

Data modeling is also used as a technique for detailing business requirements for specific databases. It is sometimes called *database modeling* because a data model is eventually implemented in a database.[4]

Topics

Data models



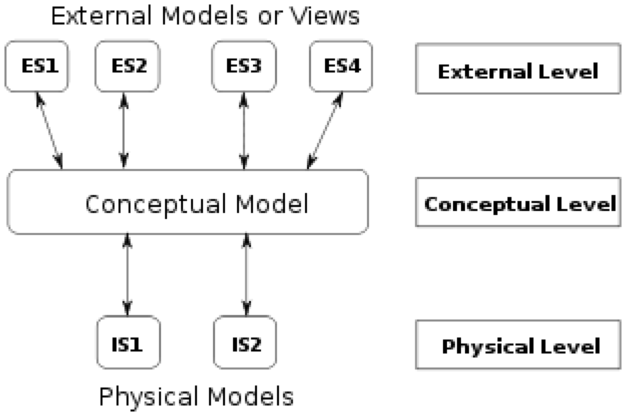
How data models deliver benefit.[1]

Data models provide a framework for data to be used within information systems by providing specific definition and format. If a data model is used consistently across systems then compatibility of data can be achieved. If the same data structures are used to store and access data then different applications can share data seamlessly. The results of this are indicated in the diagram. However, systems and interfaces are often expensive to build, operate, and maintain. They may also constrain the business rather than support it. This may occur when the quality of the data models implemented in systems and interfaces is poor.[1]

Some common problems found in data models are:

- Business rules, specific to how things are done in a particular place, are often fixed in the structure of a data model. This means that small changes in the way business is conducted lead to large changes in computer systems and interfaces. So, business rules need to be implemented in a flexible way that does not result in complicated dependencies, rather the data model should be flexible enough so that changes in the business can be implemented within the data model in a relatively quick and efficient way.
- Entity types are often not identified, or are identified incorrectly. This can lead to replication of data, data structure and functionality, together with the attendant costs of that duplication in development and maintenance. Therefore, data definitions should be made as explicit and easy to understand as possible to minimize misinterpretation and duplication.
- Data models for different systems are arbitrarily different. The result of this is that complex interfaces are required between systems that share data. These interfaces can account for between 25-70% of the cost of current systems. Required interfaces should be considered inherently while designing a data model, as a data model on its own would not be usable without interfaces within different systems.
- Data cannot be shared electronically with customers and suppliers, because the structure and meaning of data has not been standardised. To obtain optimal value from an implemented data model, it is very important to define standards that will ensure that data models will both meet business needs and be consistent.[1]

Conceptual, logical and physical schemas



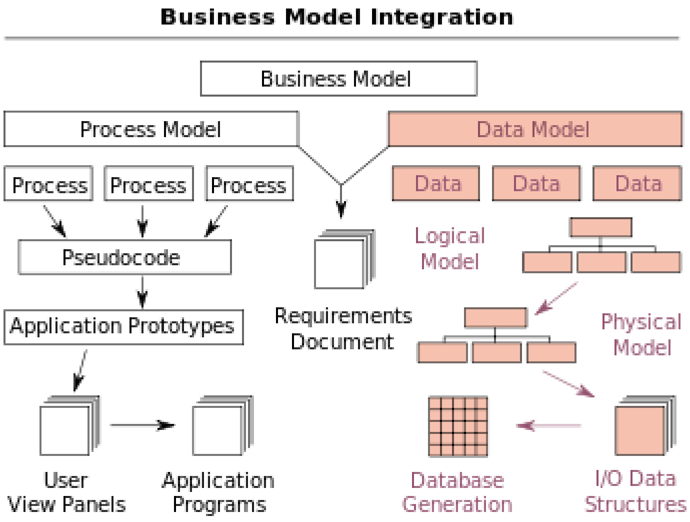
The ANSI/SPARC three level architecture. This shows that a data model can be an external model (or view), a conceptual model, or a physical model. This is not the only way to look at data models, but it is a useful way, particularly when comparing models.[1]

In 1975 ANSI described three kinds of data-model *instance*:[5]

- Conceptual schema: describes the semantics of a domain (the scope of the model). For example, it may be a model of the interest area of an organization or of an industry. This consists of entity classes, representing kinds of things of significance in the domain, and relationships assertions about associations between pairs of entity classes. A conceptual schema specifies the kinds of facts or propositions that can be expressed using the model. In that sense, it defines the allowed expressions in an artificial "language" with a scope that is limited by the scope of the model. Simply described, a conceptual schema is the first step in organizing the data requirements.
- Logical schema: describes the structure of some domain of information. This consists of descriptions of (for example) tables, columns, object-oriented classes, and XML tags. The logical schema and conceptual schema are sometimes implemented as one and the same.[2]
- Physical schema: describes the physical means used to store data. This is concerned with partitions, CPUs, tablespaces, and the like.

According to ANSI, this approach allows the three perspectives to be relatively independent of each other. Storage technology can change without affecting either the logical or the conceptual schema. The table/column structure can change without (necessarily) affecting the conceptual schema. In each case, of course, the structures must remain consistent across all schemas of the same data model.

Data modeling process



Data modeling in the context of Business Process Integration.[6]

In the context of business process integration (see figure), data modeling complements business process modeling, and ultimately results in database generation.[6]

The process of designing a database involves producing the previously described three types of schemas - conceptual, logical, and physical. The database design documented in these schemas are converted through a Data Definition Language, which can then be used to generate a database. A fully attributed data model contains detailed attributes (descriptions) for every entity within it. The term "database design" can describe many different parts of the design of an overall database system. Principally, and most correctly, it can be thought of as the logical design of the base data structures used to store the data. In the relational model these are the tables and views. In an object database the entities and relationships map directly to object classes and named relationships. However, the term "database design" could also be used to apply to the overall process of designing, not just the base data structures, but also the forms and queries used as part of the overall database application within the Database Management System or DBMS.

In the process, system interfaces account for 25% to 70% of the development and support costs of current systems. The primary reason for this cost is that these systems do not share a common data model. If data models are developed on a system by system basis, then not only is the same analysis repeated in overlapping areas, but further analysis must be performed to create the interfaces between them. Most systems within an organization contain the same basic data, redeveloped for a specific purpose. Therefore, an efficiently designed basic data model can minimize rework with minimal modifications for the purposes of different systems within the organization[1]

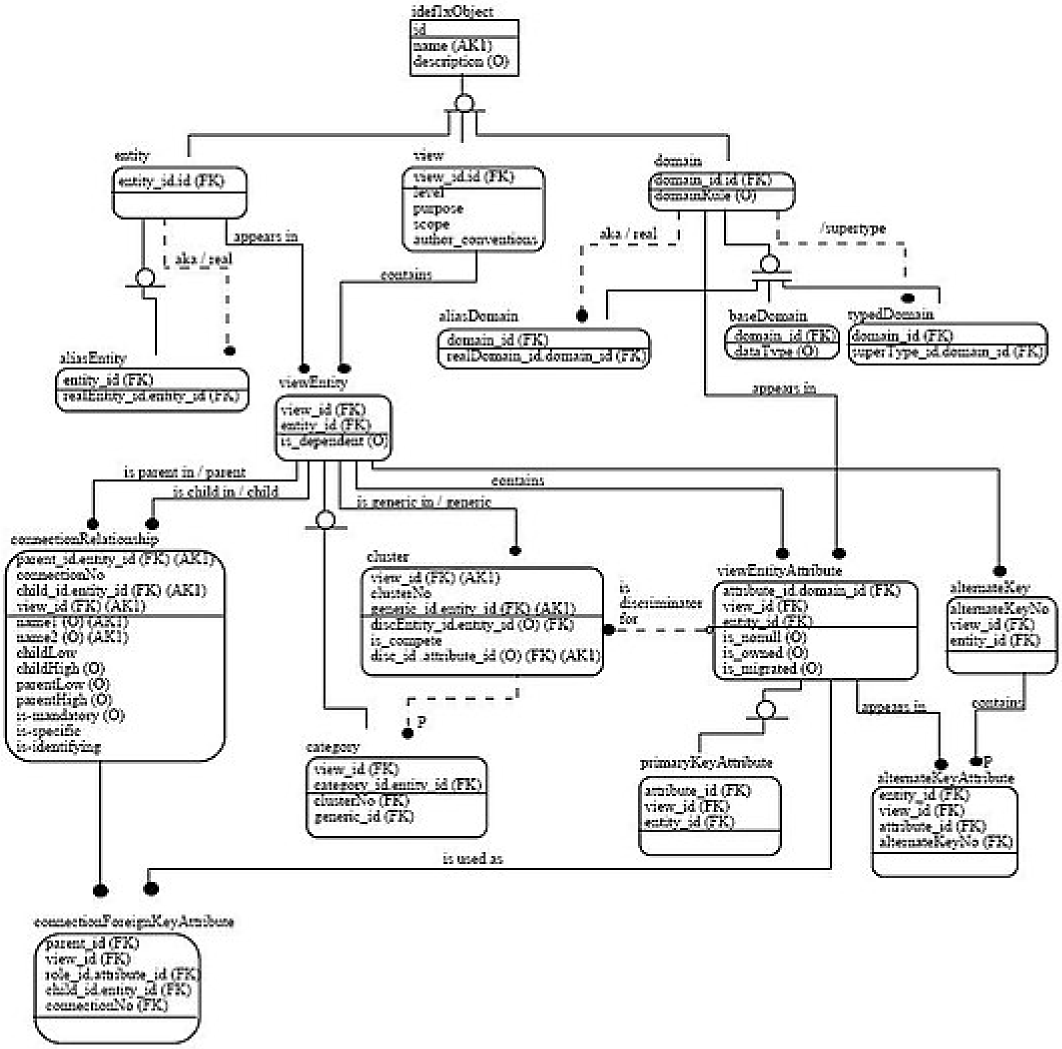
Modeling methodologies

Data models represent information areas of interest. While there are many ways to create data models, according to Len Silverston (1997)[7] only two modeling methodologies stand out, top-down and bottom-up:

- Bottom-up models or View Integration models are often the result of a reengineering effort. They usually start with existing data structures forms, fields on application screens, or reports. These models are usually physical, application-specific, and incomplete from an enterprise perspective. They may not promote data sharing, especially if they are built without reference to other parts of the organization.[7]
- Top-down logical data models, on the other hand, are created in an abstract way by getting information from people who know the subject area. A system may not implement all the entities in a logical model, but the model serves as a reference point or template.[7]

Sometimes models are created in a mixture of the two methods: by considering the data needs and structure of an application and by consistently referencing a subject-area model. Unfortunately, in many environments the distinction between a logical data model and a physical data model is blurred. In addition, some CASE tools don't make a distinction between logical and physical data models.[7]

Entity–relationship diagrams



Example of an IDEF1X entity–relationship diagrams used to model IDEF1X itself. The name of the view is mm. The domain hierarchy and constraints are also given. The constraints are expressed as sentences in the formal theory of the meta model.[8]

There are several notations for data modeling. The actual model is frequently called "entity–relationship model", because it depicts data in terms of the entities and relationships described in the data.[4] An entity–relationship model (ERM) is an abstract conceptual representation of structured data. Entity–relationship modeling is a relational schema database modeling method, used in software engineering to produce a type of conceptual data model (or semantic data model) of a system, often a relational database, and its requirements in a top-down fashion.

These models are being used in the first stage of information system design during the requirements analysis to describe information needs or the type of information that is to be stored in a database. The data modeling technique can be used to describe any ontology (i.e. an overview and classifications of used terms and their relationships) for a certain universe of discourse i.e. area of interest.

Several techniques have been developed for the design of data models. While these methodologies guide data modelers in their work, two different people using the same methodology will often come up with very different results. Most notable are:

- Bachman diagrams
- Barker's notation
- Chen's notation
- Data Vault Modeling
- Extended Backus–Naur form
- IDEF1X
- Object-relational mapping
- Object-Role Modeling
- Relational Model
- Relational Model/Tasmania

Generic data modeling

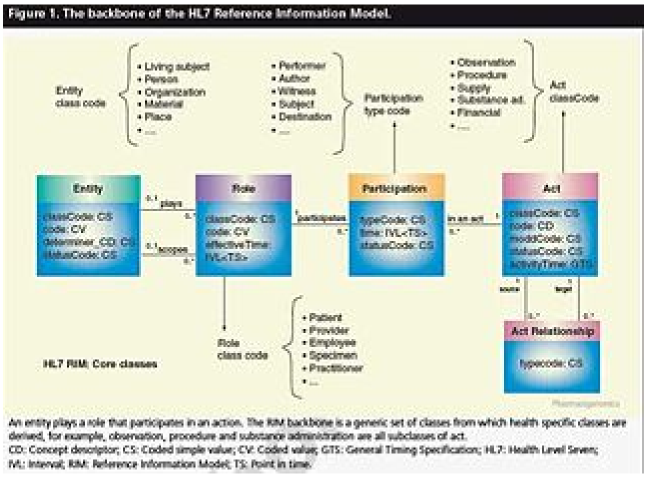


Figure 1. The backbone of the HL7 Reference Information Model.

An entity plays a role that participates in an action. The RIM backbone is a generic set of classes from which health specific classes are derived, for example, observation, procedure and substance administration are all subclasses of act.
CD: Concept descriptor; CS: Coded simple value; CV: Coded value; GTS: General Timing Specification; HL7: Health Level Seven; IVL: Interval; RIM: Reference Information Model; TS: Point in time.

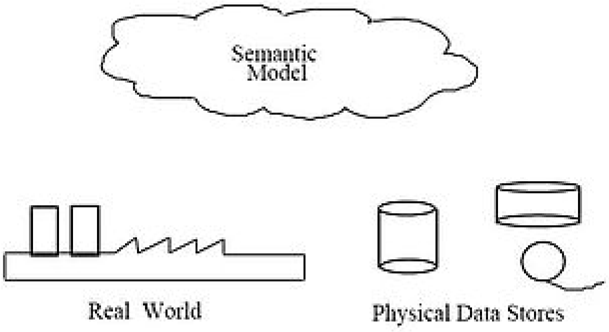
Example of a Generic data model.[9]

Generic data models are generalizations of conventional data models. They define standardized general relation types, together with the kinds of things that may be related by such a relation type. The definition of generic data model is similar to the definition of a natural language. For example, a generic data model may define relation types such as a 'classification relation', being a binary relation between an individual thing and a kind of thing (a class) and a 'part-whole relation', being a binary relation between two things, one with the role of part, the other with the role of whole, regardless the kind of things that are related.

Given an extensible list of classes, this allows the classification of any individual thing and to specify part-whole relations for any individual object. By standardization of an extensible list of relation types, a generic data model enables the expression of an unlimited number of kinds of facts and will approach the capabilities of natural languages. Conventional data models, on the other hand, have a fixed and limited domain scope, because the instantiation (usage) of such a model only allows expressions of kinds of facts that are predefined in the model.

Semantic data modeling

The logical data structure of a DBMS, whether hierarchical, network, or relational, cannot totally satisfy the requirements for a conceptual definition of data because it is limited in scope and biased toward the implementation strategy employed by the DBMS. That is unless the semantic data model is implemented in the database on purpose, a choice which may slightly impact performance but generally vastly improves productivity.



Semantic data models.[8]

Therefore, the need to define data from a conceptual view has led to the development of semantic data modeling techniques. That is, techniques to define the meaning of data within the context of its interrelationships with other data. As illustrated in the figure the real world, in terms of resources, ideas, events, etc., are symbolically defined within physical data stores. A semantic data model is an abstraction which defines how the stored symbols relate to the real world. Thus, the model must be a true representation of the real world.[8]

A semantic data model can be used to serve many purposes, such as:[8]

- planning of data resources
- building of shareable databases
- evaluation of vendor software
- integration of existing databases

The overall goal of semantic data models is to capture more meaning of data by integrating relational concepts with more powerful abstraction concepts known from the Artificial Intelligence field. The idea is to provide high level modeling primitives as integral part of a data model in order to facilitate the representation of real world situations.[10]

See also

- Architectural pattern (computer science)
- Comparison of data modeling tools

- Data (computing)
- Data dictionary
- Document modeling
- Enterprise data modeling
- Entity Data Model
- Information management
- Informative modeling
- Metadata modeling
- Three schema approach
- Zachman Framework

References

1. Matthew West and Julian Fowler (1999). Developing High Quality Data Models (https://sites.google.com/site/drmatthewwest/publications/princ03.pdf) . The European Process Industries STEP Technical Liaison Executive (EPISTLE).
2. Simison, Graeme. C. & Witt, Graham. C. (2005). *Data Modeling Essentials*. 3rd Edition. Morgan Kaufmann Publishers. ISBN 0-12-644551-6
3. Data Integration Glossary (http://knowledge.fhwa.dot.gov/tam/aashto.nsf/All+Documents/4825476B2B5C687285256B1F00544258/%24FILE/DIGloss.pdf) Archived (https://web.archive.org/web/20090320001015/http://knowledge.fhwa.dot.gov/tam/aashto.nsf/All+Documents/4825476B2B5C687285256B1F00544258/%24FILE/DIGloss.pdf) March 20, 2009, at the Wayback Machine, U.S. Department of Transportation, August 2001.
4. Whitten, Jeffrey L.; Lonnie D. Bentley, Kevin C. Dittman. (2004). *Systems Analysis and Design Methods*. 6th edition. ISBN 0-256-19906-X.
5. American National Standards Institute. 1975. *ANSI/X3/SPARC Study Group on Data Base Management Systems; Interim Report*. FDT (Bulletin of ACM SIGMOD) 7:2.
6. Paul R. Smith & Richard Sarfaty (1993). Creating a strategic plan for configuration management using Computer Aided Software Engineering (CASE) tools. (https://www.osti.gov/energycitations/purl.cover.jsp;jsessionid=6192EDBFBAB7DCED13883C55F221221A?purl=/10160331-YhIRrY/) Paper For 1993 National DOE/Contractors and Facilities CAD/CAE User's Group.
7. Len Silverston, W.H.Inmon, Kent Graziano (2007). *The Data Model Resource Book*. Wiley, 1997. ISBN 0-471-15364-8. Reviewed by Van Scott on tdan.com (http://www.tdan.com/view-book-reviews/5593) . Accessed November 1, 2008.

8. FIPS Publication 184 (http://www.itl.nist.gov/fipspubs/idef1x.doc) Archived (https://web.archive.org/web/20131203223034/http://www.itl.nist.gov/fipspubs/idef1x.doc) December 3, 2013, at the Wayback Machine released of IDEF1X by the Computer Systems Laboratory of the National Institute of Standards and Technology (NIST). December 21, 1993.
9. Amnon Shabo (2006). Clinical genomics data standards for pharmacogenetics and pharmacogenomics (https://healthit.hhs.gov/portal/server.pt?open=512&objID=1263&mode=2) Archived (https://web.archive.org/web/20090722232240/http://healthit.hhs.gov/portal/server.pt?open=512&objID=1263&mode=2) July 22, 2009, at the Wayback Machine.
10. "Semantic data modeling" In: *Metaclasses and Their Application*. Book Series Lecture Notes in Computer Science. Publisher Springer Berlin / Heidelberg. Volume Volume 943/1995.

- This article incorporates public domain material from the National Institute of Standards and Technology website https://www.nist.gov (https://www.nist.gov) .

Further reading

- J.H. ter Bekke (1991). *Semantic Data Modeling in Relational Environments*
- John Vincent Carlis, Joseph D. Maguire (2001). *Mastering Data Modeling: A User-driven Approach*.
- Alan Chmura, J. Mark Heumann (2005). *Logical Data Modeling: What it is and how to Do it*.
- Martin E. Modell (1992). *Data Analysis, Data Modeling, and Classification*.
- M. Papazoglou, Stefano Spaccapietra, Zahir Tari (2000). *Advances in Object-oriented Data Modeling*.
- G. Lawrence Sanders (1995). *Data Modeling*
- Graeme C. Simsion, Graham C. Witt (2005). *Data Modeling Essentials*'
- Matthew West (2011) *Developing High Quality Data Models*

External links

Wikimedia Commons has media related to ***Data modeling***.

- Agile/Evolutionary Data Modeling (http://www.agiledata.org/essays/agileDataModeling.html)
- Data modeling articles (http://www.softdevarticles.com/modules/weblinks/viewcat.php?cid=21)
- Database Modelling in UML (http://www.methodsandtools.com/archive/archive.php?id=9)
- Data Modeling 101 (http://www.agiledata.org/essays/dataModeling101.html)
- Semantic data modeling (http://www.jhterbekke.net/SemanticDataModeling.html)

- System Development, Methodologies and Modeling (http://www.cems.uwe.ac.uk/~tdrewry/modeling.htm) Notes on by Tony Drewry
- Request For Proposal - Information Management Metamodel (IMM) (http://www.omg.org/cgi-bin/doc?ab/05-12-02) of the Object Management Group
- Data Modeling is NOT just for DBMS's Part 1 (https://web.archive.org/web/20120625215800/http://www.ipl.com/papers/Data%20modelling%20is%20NOT%20JUST%20for%20DBMS%20part%201.pdf) Chris Bradley
- Data Modeling is NOT just for DBMS's Part 2 (https://web.archive.org/web/20120626144315/http://www.ipl.com/papers/Data%20modelling%20is%20NOT%20JUST%20for%20DBMS%20part%202.pdf) Chris Bradley