pennsylvania
OFFICE OF ADMINISTRATION

Information Technology Policy

Data Modeling Basics

Number
STD-INF003B

Effective Date
August 2, 2005

Category
Information

Supersedes
All Prior Versions

Contact
RA-ITCentral@pa.gov

Scheduled Review
September 2023

1. Definitions

1.1. Aggregation: a technique used to consolidate, collect, and present summarized Key Performance Indicator (KPI) data; one that optimizes data retrieval by summarizing rows of a fact table according to specific Dimensions or attributes.

1.2. Business Rule: stipulates a specific business-related definition or information that is linked to database objects. The information is always associated with the business facts or descriptions; or it might be formulas or algorithms, either client-based or destined for the server. Once defined, Business Rules can be applied across the enterprise and across various enterprise and analytical applications.

1.3. Cardinality: indicates the relationship for linked or associated entities (one or many) of an Entity in relation to another Entity. You can select the following values for Cardinality:

- One-to-one - One instance of the first Entity can correspond to only one instance of the second Entity.
- One-to-many - One instance of the first Entity can correspond to more than one instance of the second Entity.
- Many-to-one - More than one instance of the first Entity can correspond to the same one instance of the second Entity.
- Many-to-many - More than one instance of the first Entity can correspond to more than one instance of the second Entity.

1.4. Data Attribute: A term used in Logical Data Models (LDM) to describe a kind of fact common to all or most instances of an Entity. Student ID is an attribute of the Entity Student. The corresponding Physical Data Model (PDM) generally implements the attribute as a database column or field.

1.5. Data Element: An Entity, attribute, database table, or database column used to represent business information in Logical or PDMs. Data element primarily defines the metadata and represents data atomicity. Users should be aware that the literature also defines Data Element to explicitly mean an attribute of an Entity. However, as defined in this

document, the term encompasses both Entities and attributes in LDMs as well as tables and columns in PDMs.

1.6. Data Entity: A term used in LDMs to describe a business Entity. For example, class of persons, places, things, concepts, or events of interest to the business, about which the business intends to keep facts. The corresponding PDM generally implements the Entity in a database table or view.

1.7. Dimension: Dimensions refers to the business objects signifying nonvalue fields or attributes used to attribute the axis of investigation of a fact. The Dimensions define the significance of the KPIs or facts.

1.8. Domain: A way of identifying and grouping the types of data items in the model. This makes it easier to standardize data characteristics for attributes or columns in different Entities or tables. Some database management systems (DBMSs) will implement Domains as "user defined datatypes." Another feature of Domains is in the maintenance of similar columns. If all "name" columns (LastName, CityName, ProductName, etc.) are defined as a common Domain, changing the datatype from char(40) to char(50) is a one-step procedure, rather than having to visit each table and search for the correct columns.

1.9. Enterprise Class DBMS: integrates multiple business processes or applications into a single DBMS and hardware platform. This contrasts with creating application specific DBMSs.

1.10. Entity: A business object or concept (often a person, place, or other thing) for which information will be stored.

1.11. Inheritance: Inheritance is the process by which a child Entity can be defined or derived from another Entity (parent Entity). Typically, the characteristics of the parent and child Entities will be similar, however the child Entity will have some additional characteristics or attributes.

1.12. Logical Data Model: Data model indicating the relationships of entities, representing a structured representation of the data of importance to the business, in terms of Entities, attributes, and their Relationships including the Business Rules that govern them. The representation includes both graphical depictions and textual definitions. LDMs are used to translate business requirements into data representations that are understandable to information systems professionals.

1.13. Logical Data Name: A unique identifier of an Entity or attribute as stored within a LDM or data dictionary. Logical Data Names should consist of English words and must be understandable by the end user. Also known as the business name or functional name.

1.14. Physical Data Model: A structured representation of the data of importance to the business, in terms of database tables and columns along with their Relationships, formats, and Business Rules that govern the data. The representation includes both graphical depictions and textual definitions. PDMs are used exclusively by information systems professionals to deploy database systems using appropriate database software.

1.15. Physical Data Name: A unique identifier of an Entity or attribute as implemented within one or more database systems. Physical Data Names are generally constrained by the limitations of the database software.

1.16. Referential Integrity: Referential Integrity refers to rules governing relationships between two or more tables. Relationships between entities in different tables are achieved using primary and foreign keys. Referential Integrity ensures that that a valid link is maintained between the primary key in the one table and the foreign keys in associated tables. It also enforces those subsequent updates or deletions do not cause a mismatch between the tables.

1.17. Relationship: A Relationship is an association or connection between Entities in a data model. Relationships are established based on Business Rules and can vary in terms of Cardinality and optionality. Relationships are modeled as a line connecting the entities.

2. Why is Data Modeling Important?

Data modeling allows the definition of the common business entities, their Relationships, and uses them to define a standardized common enterprise model.

Data modeling is a vital part in the development process. One can compare this to creating a blueprint to build a house before the actual building takes place. As much as the blueprint takes time to prepare, it can also go through multiple iterations of validation to ensure the foundation, structure, and aesthetics of the building plan conform to intended objectives and quality standards. Therefore, data modeling is an intensive process which consumes a major part of the development time.

The model is built in a phased manner and goes through several iterations of validation to ensure that the structure and content of the model addresses the business objectives of the enterprise or application, meets quality standards, is modular, provides a solid foundation for future extensions, and data reuse for other enterprise applications. Less time spent Data Modeling will only produce a weak unstructured model that will be expensive to maintain in the future, may produce inconsistent results, incorrect results, and will be unfit for reporting or future extensions.

Major events in data modeling include:

- Identifying business requirements, Domains and business entities.
- Identifying the Entity Relationships and Cardinality.
- Data definitions Segregation – Dimensions and attributes, individual entities, KPIs, measures, and derived entities such as formulas.
- Identifying Entities, data requirements, and processes.
- Defining attributes of the data such as data types, sizes, and defaults.
- Applying validation and Business Rules to ensure data integrity.
- Defining data management and security processes.
- Specifying data archival and storage.

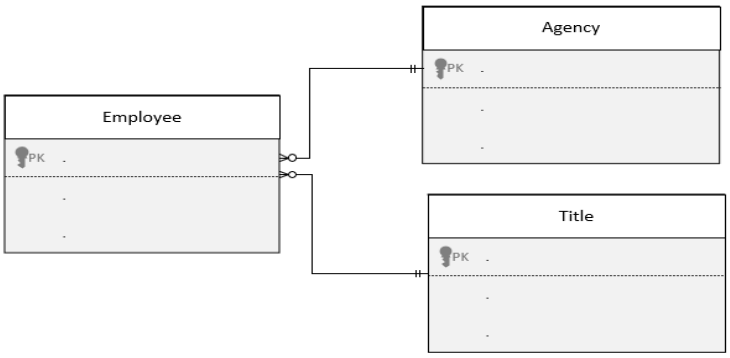
3. How are Data Models Used?

The three most common types of data models are Conceptual, Logical, and Physical data models. Each type of model has a distinct purpose and audience. Data modeling is generally an iterative process with the three models being created in order. The Conceptual data model (CDM) which has a higher level of abstraction is considered the starting point. The models then get more detailed, complex, and concrete as you progress through the LDM and Physical Data Model (PDM). The PDM is the last stage of the process and serves as the blueprint for the physical construction of the data base. A summary of each type of model is provided below:

3.1 CDMs.

A CDM is a high-level visual representation of data and the Relationships between the data. These models establish the entities and the Relationship between entities. This type of model is ideal for conveying information to stakeholders as it illustrates business concepts and requirements and is less technical and detailed in nature that other types of models. CDMs often serve as a starting point and can later be used to create more detailed or complex types of models.

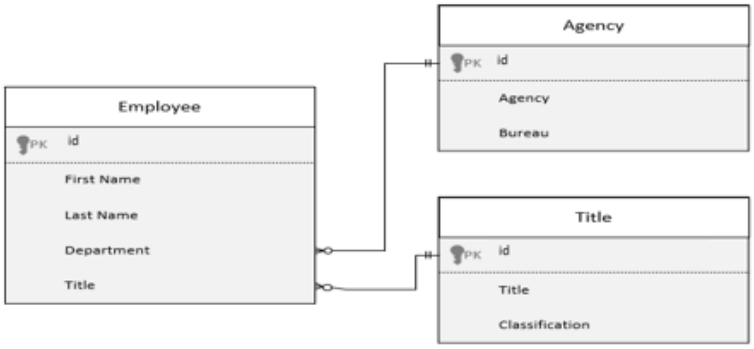
Sample CDM:



3.2 LDMs

A LDM builds upon the concepts and Relationships established in the CDM and adds an additional level of detail. The details include, the structure of data entities, data rules, mappings, Data Attributes, and other context. The audience for LDMs tends to be database analysts, designers, and architects. The LDM is still a higher-level view than the physical model.

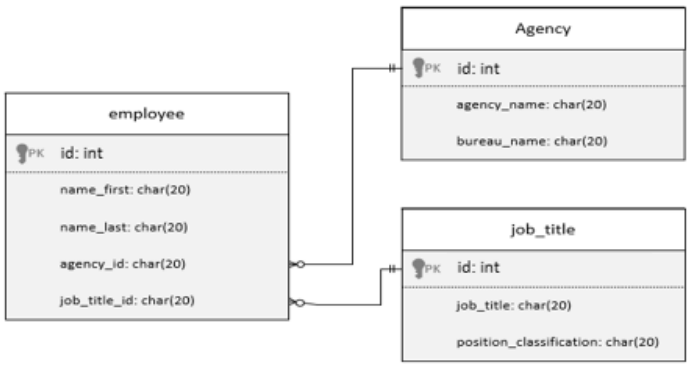
Sample LDM:



3.3 PDMs

A PDM expands upon the previous two model types by incorporating information around the physical implementation, structure, technical requirements, and performance of the database. It included details such as the schema of the database, data tables, columns, and naming conventions. PDMs will be used for the actual implementation of the database.

Sample PDM:



Data models can range in scale from an individual project or system to an entire enterprise. Enterprise level data models are especially important when there are many data sources spread across an entire enterprise. These models provide a single formalized view of data across the organization and help reduce data redundancy, provide common data definitions, and improve accuracy and interoperability. Establishing an enterprise level data model will also be beneficial for future projects as well as data warehouse and data inventory initiatives.

4. Data Modeling vs. Class Modeling:

The focus of data models is data and the definition of the common business entities. These models deal with entities and the Relationships between entities. Data models are concerned specifically with the implementation of a database rather than general system design.

A class model on the other hand, focuses on the objects in a system and the functional Relationship between those objects. While data is included in class modeling it is not the sole focus. Class models are especially useful for object-oriented designs.

5. Data Perspectives:

Data Models are a valuable source of information, providing a graphical depiction of data at different levels of abstraction. For example, the owner of a business process is interested in the conceptual view of data (CDM). The designer of the data is interested in the logical view (LDM). This view is sometimes referred to as the transformation layer. The data administrator is typically concerned with this model. The database administrator is typically more concerned with the physical implementation of a relational database (PDM).

This table summarizes the different perspectives.

Model	**Perspective**	**Model Description**	**Type of Model**	**Entity**	**Type of Relation**
Business Model	Owner	Semantic Model	Conceptual	Business Entity	Business
System Model	Designer	Logical Data Model	Logical	Data Entity	Data

Model	Perspective	Model Description	Type of Model	Entity	Type of Relation
Technology Model	Builder	Data Design	Physical	Table/ Segments	Key/Pointer
Detailed Representation	Developer	Data Definitions	N/A	Field	Address

6. Data Modeling Standards Supported:

There are three common data modeling notations: Information Engineering (IE), IDEF1X, and the Unified Modeling Language (UML).

From a notation standard, current product standards should support both IDEF1X and IE modeling standards as well as naming standards that allow you to create glossaries of approved words and enforce the way words are used in naming tables and columns.

XML (which has a very loose industry standard) should also be supported in the tool. It is likely that UML based tools will co-exist with other data modeling notations for the near future. UML provides all the syntax needed to perform data modeling, as well as behavioral modeling. Some developers see an advantage to using one modeling language for all modeling purposes.

7. Data modeling notations:

Notation	Information
IE	Several variations of this method exist as it has been adopted and modified multiple time since its creation, however it is still one of the most popular notations. The popularity of IE is due to its simplicity and flexibility. This notation is supported by most data modeling software platforms and is most commonly used for logical models.
IDEF1X	This notation has been used extensively in the armed forces and Department of Defense projects. It is considered more complex and as having more rigid rules than other notations. This notation is primarily used for logical and physical models. This notation is considered less than ideal for non-relational and object-oriented implementations.
UML	Unified Modeling Language (UML) is more general purpose in nature than other notations and can be applied to overall system design and software engineering. This notation is especially well suited for object-oriented designs.

Other notations exist, but the current standards, listed in STD-INF003A *Data Modeling Products and Standards,* either support the use of IE and IDEF1X notations or UML.

8. Model Reuse:

Creating reusable models should be a goal of any data modeling effort. Having well designed models with reusable components can save a tremendous amount of time in future projects. In practice, it also facilitates the physical sharing of data and other efficiencies across the enterprise. Alternatively, poorly designed models can lead to redundant and poor-quality data and a greater amount of time spent on future initiatives.

9. Model Review Process:

A review process with appropriate team members and stakeholders should follow each stage of model development. This process will ensure that the respective data model correctly and accurately represents the business requirements and maximizes data integrity.

NOTE: A separate effort is currently underway to develop a formal data modeling methodology, including templates and checklists to validate LDMs and PDMs.

10. Data Modeling Best Practice Standards Supported:

A list of Data Modeling Best Practices has been compiled by the Office of Data & Digital Technology. These standards have applicability across all current standard products and are required to be used for all application development efforts of sufficient size and scope. If a specific standard applies only to mission- critical applications, it will be identified as such. Reference BPD-INF003C *Data Modeling Best Practices* for the latest version.

11. Data Modeling Basic Steps

Pre-requisite:

- Business Vision and Business process modeling
- Define business entities and conceptual model

11.1 Identify Entity Types

Once business requirements are established and the entities and conceptual models have been defined, the next step is to identify the Entity types. The Entity types essentially serve as a template for grouping individual entities based on some similar properties between the entities. Once identified, the Entity types will establish the top-level structure of the data model. Entity types generally correspond to individual database tables.

11.2 Identify Attributes for each Entity Type

Once Entity types have been established, the next step is to further define our data model by identifying the attributes for each Entity type. An attribute is essentially a specific piece of information that will be stored for each Entity. Whereas the Entity type represents the data table, each attribute would represent a column in that table. For example, if we have an EMPLOYEE Entity, EMPLOYEE_NAME and EMPLOYEE_ID_NUMBER are examples of potential attributes. Only attributes that are required or of value to the business should be included.

11.3 Establish/Apply Data Naming Conventions:

Standard naming conventions should be established and applied uniformly across an individual data model or even across an enterprise when possible. These naming conventions should be adhered to both during the initial creation of the data model and during any subsequent updates. The naming conventions applied should be descriptive and easily understandable.

11.4 Identify Relationships:

The next step in the process is to identify the Relationships between Entities. In this step it is important to choose the right attributes that will be used to model the Relationships. This should also reflect the type or nature of the

Relationship:

- One-to-one
- One-to-many
- Many-to-one
- Many-to-many

11.5 Assign Keys:

Keys are attributes that identify a record in a particular table and are essential to establishing Relationships between tables. The first step is to identify candidate keys (keys that have the potential to serve as the primary key) for each table. From this list of candidate keys, a primary key can be chosen that will uniquely identify each record. Only one primary key can exist per table, and it cannot be null. For example, in a table of employees, employee ID could serve as the primary key. Foreign keys are then used to connect tables. Relationships are established between primary keys in one table and the foreign key in another.

11.6 Normalize Data:

Normalization is the process of organizing data models in such a way that data redundancy in minimized. Essentially, attributes that are applicable to more than one Entity type or table are linked using keys rather than being repeated in multiple tables. Benefits of normalization include reducing the amount of storage needed and lessening the chances of conflicting data or anomalies.

11.7 Optimize Performance:

There is a tradeoff that exist between data normalization and performance. Although normalization reduces storage space and redundancy, it can result in less efficient performance. Specifically, having to join many connecting tables results in slower queries and a greater strain on system resources. This is especially true when high transaction volumes or large complex queries are required. At times, it may be necessary to denormalize certain portions of a database to allow for faster more responsive performance. Denormalization should only be applied where performance improvements are necessary.

This chart contains a history of this publication's revisions.

Version	Date	Purpose of Revision
Original	08/2/2005	Base Document
Revision	11/18/2010	ITP Refresh
Revision	05/13/2021	ITP Refresh
Revision	09/09/2022	ITP Refresh Rewrote all sections utilizing copyrighted content