Data Modeling Standards

> **Source File:** generated-documents\dmbok\data-modeling-standards.md
> **Generated:** 30/07/2025 at 06:57:40
> **Generated by:** Requirements Gathering Agent - PDF Converter

Data Modeling Standards Guide

Generated by adpa-enterprise-framework-automation v3.2.0
Category: dmbok
Generated: 2025-07-17T22:43:33.516Z
Description: Comprehensive guide to data modeling standards, conventions, and best practices.

Data Modeling Standards Guide

1. Introduction

1.1 Purpose and Scope

This guide establishes comprehensive standards and best practices for data modeling activities throughout the data lifecycle, as aligned with the DAMA Data Management Body of Knowledge (DMBOK). It covers the creation, maintenance, and usage of conceptual, logical, and physical data models across all projects, ensuring consistency, quality, and reusability of data assets.

1.2 Intended Audience

- Data Modelers and Architects
- Database Designers and Administrators
- Business Analysts
- Application Developers
- Project Managers
- Data Stewards

1.3 Document Conventions

- **Bold**: Section headings and key terms
- *Italics*: Examples and notes
- [UPPERCASE]: Placeholders to be replaced with context-specific values
- All diagrams utilize standardized notation, with legends provided

---

2. Data Modeling Standards

2.1 Naming Conventions

Entities/Tables:

- Singular nouns (e.g., `Customer`, `Order`)
- PascalCase or Upper_Snake_Case (e.g., `OrderDetail`, `ORDER_DETAIL`)
- No spaces or special characters; use underscores if required

Attributes/Columns:

- LowerCamelCase or lower_snake_case (e.g., `orderDate`, `order_date`)
- Descriptive and unambiguous (e.g., `customerName`, not `name`)

Relationships:

- Descriptive names using related entities (e.g., `CustomerHasOrder`)

Keys & Constraints:

- Prefixes: `PK_` for primary keys, `FK_` for foreign keys, `IX_` for indexes
- Example: `PK_Customer`, `FK_Order_CustomerId`, `IX_Order_OrderDate`

2.2 Abbreviation Standards

- Abbreviations only when necessary, using a published, project-wide abbreviation list
- Avoid truncation that causes ambiguity (e.g., `Amt` for Amount is acceptable)
- No abbreviations for entity or attribute names unless widely accepted (e.g., `ID`)
- Consistency is mandatory

2.3 Data Types and Domains

- Use standardized logical data types (e.g., `String`, `Integer`, `Decimal`, `Boolean`, `Date`)
- Map logical types to physical data types according to database platform standards (see Appendix)
- Define domains for reusable attribute sets (e.g., EmailAddress, PhoneNumber)
- Specify length, precision, and scale explicitly

2.4 Null Handling

- Default: Attributes are nullable unless business rules dictate otherwise
- Mandatory attributes must be specified as NOT NULL
- Avoid storing “magic values” (e.g., -1, 9999) for missing data
- Document nullability in model and metadata

2.5 Modeling Notation Standards

- Preferred: Crow's Foot (IE) for Entity-Relationship Diagrams (ERD)
- IDEF1X notation may be used for complex models or government projects
- UML Class Diagrams for object-oriented contexts
- All diagrams must include a legend, cardinality, and clear relationship labeling

---

3. Conceptual Data Model

3.1 Entity Definitions

- Entities represent real-world objects or concepts
- Each entity must have a clear, unambiguous definition documented in the glossary
- Example: *Customer: An individual or organization that purchases products or services.*

3.2 Relationship Rules

- Only significant business relationships are modeled
- Relationships must specify cardinality (e.g., 1:1, 1:M, M:N)
- All relationships are named and have clear definitions

3.3 Business Rules

- All business rules affecting data structure (e.g., "Each Order must be placed by exactly one Customer") must be documented
- Rules should be traceable to source requirements

3.4 Example Diagrams

(See Appendix for sample conceptual ERD diagrams.)

---

4. Logical Data Model

4.1 Attribute Definitions

- All attributes are fully named and defined
- Data type, domain, length, nullability, and default values specified
- Document business meaning for each attribute

4.2 Primary and Foreign Keys

- Every entity must have a primary key (natural or surrogate)
- Foreign keys must reference parent entity's primary key
- Composite keys allowed only if justified and documented

4.3 Normalization Rules

- Logical models must be normalized to at least 3NF (Third Normal Form)
- Exceptions require explicit business justification

4.4 Denormalization Guidelines

- Denormalization is permitted for performance or reporting needs
- All denormalized structures must be documented with rationale and impact analysis

---

5. Physical Data Model

5.1 Table Design

- Table names follow entity naming conventions
- Columns match logical attribute names, with physical naming conventions applied
- Surrogate keys should be numeric, auto-incremented

5.2 Indexing Strategy

- Primary keys and foreign keys are indexed by default
- Additional indexes based on query performance analysis
- Naming conventions for indexes: `IX_[Table]_[Column]`

5.3 Partitioning Guidelines

- Partition large tables based on access patterns (e.g., range or list partitioning)
- Document partitioning scheme and rationale

5.4 Storage Parameters

- Specify storage options (e.g., tablespace, filegroups) per platform best practices
- Define growth, retention, and archiving strategies

---

6. Model Management

6.1 Version Control

- All models are version-controlled using the enterprise's standard repository (e.g., Git, SVN)
- Each version must be tagged with date, author, and change summary

6.2 Change Management

- All changes subject to formal review and approval
- Change requests documented, including impact analysis

6.3 Model Validation

- Validate models for structural integrity (e.g., orphan entities, referential issues)
- Peer review required before promotion to production

6.4 Documentation Requirements

- Each model must include:
  - Entity and attribute definitions
  - Business rules
  - Diagrams with legends and notations
  - Revision history
  - Data lineage where applicable

---

7. Tools and Technologies

7.1 Modeling Tools

- Use approved tools (e.g., ER/Studio, ERwin, PowerDesigner, IBM Data Architect)
- Tool selection based on project and enterprise standards

7.2 Reverse Engineering

- Reverse engineering from existing databases requires review and alignment with standards
- All reverse-engineered models must be documented and cleaned

7.3 Forward Engineering

- Use model-driven generation of DDL scripts where possible
- Scripts reviewed for compliance with physical standards

7.4 Model Comparison

- Use tool-supported model diff and merge functions
- Resolve conflicts per change management process

---

8. Appendices

8.1 Glossary

- Definitions for all entities, attributes, and terms used in the models

8.2 Naming Convention Examples

Type	Example
Entity	Customer, ProductOrder
Attribute	orderDate, customerName
PK	PK_Customer
FK	FK_Order_CustomerId
Index	IX_Order_OrderDate

8.3 Common Patterns

- Master-detail relationships (1:M)
- Many-to-many resolution tables (associative entities)
- Slowly Changing Dimensions (for Data Warehousing)

8.4 Anti-patterns to Avoid

- Using reserved words as names
- Redundant data fields
- Unnamed or ambiguous relationships
- Overuse of surrogate keys without business justification
- Storing multiple values in a single column (violation of atomicity)

References:

- DAMA-DMBOK2: Data Management Body of Knowledge, 2nd Edition

- Project-specific data governance policies
- Platform/database vendor documentation

Contact:

For questions or clarifications, contact the Data Modeling Lead or Data Governance Team.
