Nationwide Confidential

Data Modeling

Best Practices

Nationwide®
On Your Side™

TABLE OF CONTENTS

1. Executive Overview

The purpose of this document is to provide a set of enterprise-wide Best Practices for Data Modeling at Nationwide. The Nationwide enterprise includes all Nationwide entities regardless of geographic location, line of business or channel of distribution.

Best Practices are comprised of statements that define what *should* be done (Guidelines) and what *could* be done to contribute to the highest, most resource-effective performance of a discipline. Best Practices defining what *should* be done are guidelines for tasks in the data modeling space. Best Practices defining what *could* be done present alternatives that are acceptable in some situations.

This document addresses relational and dimensional data modeling in general without regards to specific tools or type of database. This version focuses on Best Practices with procedures to be addressed at a later time. These Best Practices were based upon a broad range of experience and knowledge by Nationwide Data Modelers and Architects across the enterprise. Existing Standards and Best Practices were reviewed and incorporated where appropriate.

The Best Practices within this document are intended to be used on design and development of new databases. Maintenance to existing data models should follow these Best Practices as much as possible realizing that concessions may need to be made for purchased packages or existing databases where different standards had been used when originally created (grandfather clause).

The Best Practices are defined by a number used for easy reference and tracking purposes. Each Best Practice has been assigned a unique number in chronological order by which they were developed. Thus, Best Practices may not appear in numeric order as new Best Practices can be inserted between old Best Practices. Each Best Practice has a short and long description statement and rationale followed by supporting statements as needed.

2. Concepts

There are different types data models. Each type contains a unique perspective to be communicated.

A Conceptual Data Model is used to communicate a broad view of business groupings important to the business and the relationships between those groupings. The context of the conceptual data model may encompass the whole of the business or a subject area within the business. Reference models such as IAA should be used to create Conceptual Data Models.

A Logical Data Model represents a business view, but at a lower level. A Logical Data Model communicates a more detailed view of the business groupings and their relationships. It also includes information needed by the business that is specific to each business grouping. The Logical Data Model is a key component to developing a Physical Data Model.

A Physical Data Model includes data that meets the needs of the business while considering specific needs of the application system and the data store platform on which the data will reside. The table below outlines key differences between the types of models. In addition to the information contained in this document, information regarding these types of models and their role in project delivery can be found on the ESDM website.

Characteristic:	**Conceptual**	**Logical**	**Physical**
Perspective: The viewpoint represented by the model	Generalized Business Groupings and Highest-Level Relationships	Detailed Business Objects and Lowest-Level Relationships	Physical Data Components
Audience	Project Managers, Solution Analysts, Project Architects	IT Design Team and Architects	Technical Subject Matter Experts, DBAs, Developers
Terminology Used	Standard Business Terms modified by abstractions that normalization introduces.	Standard Business Terms with technical terms as needed.	Technical Terms, abbreviations, special words and syntax imposed by the Database Management Systems in use
Database	Independent	Independent	Dependent
Platform	Independent	Independent	Dependent
Normalization	Normalized *(IAA should be leveraged)*	Normalized	Normalized; May de-normalize for documented performance reasons with exception approval.

Characteristic:	Conceptual	Logical	Physical
Components	Entities and Relationships	Entities, Attributes, and Relationships	Tables, Columns, Relationships, Constraints, and Indexes
Deployment	Not deployed because of the level of abstraction employed.	Deployable in a physical design (domain model, XML, physical database).	Deployed.

3. Best Practices

3.1 Development Guidelines

Logical Data Model

Data Modeling Guideline #G1: Identify entities and attributes that are not physically implemented. **Description:** All entities and attributes that are not included in the physical data model shall be marked as "Logical only" in the data modeling tool, if available. **Rationale**: Indication of foresight and placement of identified data reflects that the implementation project was not able to keep the data in scope, but records the need for the data in a future release. Capturing information at the time of analysis and design reduces re-work needed in a later phase. Indicating the column was not physically implemented reduces confusion when comparing the logical to physical data model.	**Source:** Data Modeling Working Group **Status:** Working Group Approved

Data Model Guideline #G2: Domains should be used to standardize common data types in attributes. **Description:** Attributes should be categorized with common properties and purposes, such as key, code, date, indicator, amount, name, description. **Rationale**: Common properties make the content of the data element more intuitive and easier to share between systems.	**Source:** DCDirect-Data-12 and NF Logical Data Modeling and Object Naming Standards **Status:** Working Group Approved

Data Modeling Guideline #G3: Verb phrases should be used to document relationships. **Description:** Use meaningful verb phrases to describe the business rules that define the constraints and relationships between entities. The verb phrase should be meaningful to the business and should be in the active form. **Rationale**: Complete documentation of business rules that were used to design the data solution is needed to avoid misunderstanding, obtain validation by the users of the design, and documentation to define the scope of the project.	**Source:** NF Logical Data Modeling and Object Naming Standards **Status:** Working Group Approved

Data Modeling Guideline #G4: Many to many relationships should not exist in the logical data model. **Description:** Resolve all many-to-many relationships. If the business rules indicated that such a relationship could exist, an associative entity between the two entities may be needed. **Rationale**: In a completed, normalized, logical model, there will not be any many-to-many relationships; whereas during the development process, many-to-many relationships will be present. Many-to-many relationships	**Source:** NF Logical Data Modeling and Object Naming Standards **Status:** Working Group Approved

indicate that enough thought has not gone into the database design.	

Physical Data Model

Data Modeling Guideline #G5: Surrogate keys should be used as primary keys on tables. **Description:** A single-part, artificially established identifier for an entity is recommended for the primary key with appropriate indexes to be created on the physical design to address unique columns that will make a given row unique. **Rationale**: Surrogate keys on tables guarantee uniqueness, support unanticipated business changes, and support tables where a unique natural business key does not exists at the time of data entry. Even when a unique natural key exists at the point of creation, such keys may be formatted, updated, deleted, recycled or reused according to the dictates of production over time. Surrogate keys allow for easier maintenance of the database over time. **Exception:** Databases where it has been proven that performance suffers when enforcement occurs at the database as compared to the ETL or application code.	**Source:** PCIT AD1.2.2 Surrogate Key and DCDirect-Data-6 **Status:** Working Group Approved

Referential Integrity

Data Modeling Guideline #G6: An architecture strategy for enforcing referential integrity and maintaining concurrency control should be planned and implemented. **Description:** A Data Quality strategy shall encompass all tiers of the application. The strategy may consist of, but not be limited to enforcing referential integrity through the application and combination of Business Rules, Database Constraints, Database Triggers, Meta Data, and Updateable Views. **Rationale**: Due to limitations/features of various DBMS/Appliance options, a single statement of direction cannot be stated. The critical point is that a strategy should be planned and common objects/databases utilized where such objects fit.	**Source:** Data Modeling Working Group **Status:** Data COE Approved

Data Modeling Guideline #G7: Natural keys should be noted in the Physical Data Model. **Description:** Columns that uniquely identify an occurrence within a table should be documented in the physical data model. Unique indexes can be built to support the natural key. **Rationale**: The physical data model evolves into a script that creates the database structures including indexes used to quickly access data. This also serves as a means of enforcing integrity through constraints that govern the relationship between one table to one or more other tables. **Implementation:** Features of data modeling tools may handle the display differently. Tools that identify unique indexes (thus the combination of attributes making a unique row) may have capabilities in them to indicate they make up a unique row; i.e., Erwin uses AK#.# to identify the order within a unique index. A simpler tool may not have that ability in which case changing the font to bold or italics could be used. A legend on the model should be used to explain the notation.	**Source:** Data Modeling Working Group **Status:** Data COE Approved.

Data Validation at the Database Level

Data Modeling Guideline #G8: Use check constraints wherever possible. **Description:** Any business rule which can be enforced via a check constraint should have one created. **Rationale**: It is more efficient to enforce data rules at the database level. Ensuring all applications and access methods that potentially create and maintain data follow the rule implemented on the database reduces maintenance efforts in multiple locations as business rules changes as well as enforcing the integrity of the contents of the database. An example of when a check constraint would be used is when a business rule states the only valid values for a flag should be "Y", "N" or a space. **Implementation (how):** Data with complex business rules should be validated very carefully to make sure there is only a single point of entry. Use of most data modeling tools allows for the creation of check constraints at the column level. **Exception:** Databases where it has been proven that performance suffers when enforcement occurs at the database as compared to the ETL or application code.	**Source:** DCDirect-Data-9 **Status:** Working Group approved.

Defaults

Data Modeling Guideline #G9: Defaults should be applied to columns defined that do not allow NULLS. **Description:** If a default value that can be defined at the database level is known, it should be set for columns that do not allow for NULL values within the data modeling tool. If the default value does not exist for the column that cannot contain NULLs, a common default, such as spaces, zeroes, high dates, etc. should be used. **Rationale**: When a common value is used for a given column, it is more efficient for the database to populate the column than the application. When new columns that cannot contain NULLS are added to an existing table, the default value allows for successful execution of the UPDATE.	**Source:** DCDirect-Data-10 **Status:** Working Group Approved

Foreign Key Index

Data Modeling Guideline #G10: When supported by the given DBMS, foreign key columns should be indexed. **Description:** Indexes should be created for all foreign keys unless the chosen DBMS does not allow it. The data modeling tool should generate a corresponding constraint on the foreign key, if needed. **Rationale**: Performance of joins will be enhanced by using the foreign key index. Corresponding constraints created as part of the foreign key will enforce data integrity of the relationship between the two tables. **Implementation**: Foreign keys shall be documented within the physical database where supported. In unusual instances where the database may not support the index, the reason for the exception should be documented on the column which was not indexed.	**Source:** DCDirect-Data-14 **Status:** Working Group Approved

Natural Key Index

Data Modeling Guideline #G11: All columns of the natural key shall be indexed by a unique key index. **Description:** When supported by the chosen DBMS, the physical data model should ensure that a unique index is created for the columns that comprise the natural key. **Rationale**: Database performance is improved when indexes are used to access and join tables for which there is a natural key. **Exceptions:** Exceptions should only occur in unusual cases and the reason for the exception should be documented on the column(s) which were not indexed.	**Source:** Data Modeling Working Group **Status:** Working Group Approved

3.2 Object Naming Standards

General Naming Standards

Data Modeling Guideline #G12: Names should be in singular form. **Description:** Use singular form when naming data objects. **Rationale:** Recommended by ISO in ISO/IEC 11179-5. Promotes naming consistency within database diagrams. Makes people use correct grammar. It's much better grammatically to say "The product table contains products" than "The Products table contains Products". Promotes consistency with naming of primary key fields and lookup tables. For example, Product table has Product_ID, Employee has Employee_ID. **Exception:** When the concept itself is plural.	**Source:** Data Modeling Working Group **Status:** Working Group Approved

Logical Naming Standards

Data Modeling Guideline #G13: Logical names should contain capitalized, whole words using spaces between words. **Description:** Logical names shall consist of one or more spelled-out words. Each word should begin with a capital letter. A space should be between words. An acronym is acceptable as a word in an entity or attribute name as long as the acronym is globally understood and documented in the Nationwide Data COP Acronym List. **Rationale**: Promote understanding and consistency of logical names. **Exceptions:** Size limitations of the data modeling tool may dictate the need to use abbreviations.	**Source:** Data Modeling Working Group **Status:** Working Group Approved

Data Modeling Guideline #G14: Verbs (if any) should be in the present tense. **Description:** **Rationale**: Recommended by ISO in ISO/IEC 11179-5. Promotes naming consistency with database diagrams. **Exception:** Other tenses are permitted if used as an adjective.	**Source:** Data Modeling Working Group **Status:** Working Group Approved

Data Modeling Guideline #G15: Avoid use of articles, prepositions, and conjunctions. **Description:** Logical names should not contain articles (a, an, the), prepositions (at, by, for, from, in, of, to, etc.), or conjunctions (and, or, but). **Rationale**: Use of prepositions and conjunctions indicates incomplete data modeling. **Exception:** The article or preposition clearly aids in identifying an information requirement term commonly used in the business or adds clarity to the name.	**Source:** Data Modeling Working Group **Status:** Working Group Approved

Data Modeling Guideline #G16: Avoid the use of possessive forms. **Description:** Logical names should not contain the possessive forms of a word, i.e., a word that denotes ownership. **Rationale**: Possessive form may indicate proper analysis and normalization has not been performed.	**Source:** Data Modeling Working Group **Status:** Data COE Approved

Data Modeling Guideline #G17: Avoid the use of formal names or organizations. **Description:** Logical names should not reference formal names or organizations. This includes computer or information systems names, hardware, software, department, delivery system, forms, screens or reports. **Rationale**: This is indicative of problems with the data design at the logical level. The physical data model may include formal names.	**Source:** Data Modeling Working Group **Status:** Data COE Approved

Entities

Attributes

Data Modeling Guideline #G18: Attributes should include a prime word, qualifier, and classword. **Description:** Formatting of attributes should always include a prime word and qualifier. In most instances a classword should be included at the end. **Rationale**: The consistency of a common format of attributes names makes the data model easier to use and understand.	**Source:** Data Modeling Working Group **Status:** Data COE Approved

Data Modeling Guideline #G19: Spell out entire attribute names. **Description:** The logical data model should have fully spelled out attribute names. Acronyms may be used if they are recognized as industry standard and documented in the Enterprise Data COE Acronym List. For long names that may exceed tool limitations, abbreviations may be used if they are identified on the Enterprise Data COE Abbreviation list. **Rationale**: Names in the logical data model need to be meaningful and easy to understand. Names that need to be shortened should be abbreviated consistently from model to model.	**Source:** Data Modeling Working Group **Status:** Data COE Approved

Physical Naming Standards

Data Modeling Guideline #G20: Table and column names should be spelled out completely when possible with abbreviations used when necessary. **Description:** Spell out words in their entirety when possible. When database limitations for name length is met, use abbreviations from the Data COP Standard Abbreviation List. Use of abbreviations should be consistent within a table. **Rationale**: Names need to be meaningful and easy to understand. Consistency of abbreviation use on a table will make it easier for consumers of the data to identify columns. For example, if a column in a table abbreviates a word in one place, every column in the table with that name component should use the same abbreviation. **Exception:** An exception to this guideline is made for industry recognized acronyms or abbreviations. When the abbreviation is already used in the existing database that is being altered, the abbreviation currently in use shall be used for new tables and columns.	**Source:** Data Modeling Working Group **Status:** Data COE Approved

Data Modeling Guideline #G21: The table and column names should reflect the intent of the corresponding logical names. **Description:** Names used in the physical data model should be reflective of the corresponding names used in the logical data model names. **Rationale**: Keeping table and column names the same as logical entity and	**Source:** Data Modeling Working Group **Status:** Data COE Approved

attribute names allows for intuitive mapping between logical and physical data models. **Exception:** Physical object naming standards based on the target platform for the physical data store may require additional naming constraints that deviate from this Guideline.	

Tables

Data Modeling Guideline #G22: Avoid special characters or all numbers in table names when possible. **Description:** Special characters should not be used in a table name. A tablename should be constructed of meaningful name that rarely takes the form of all numbers. If a number is required as part of the name, it is best to spell out the numbers. **Rationale**: Special characters are often prohibited in many RDBMS. **Example**: A table name of *1099* should contain other words to describe the data about the 1099 that is stored. A table name of *1* provides no business or technical value.	**Source:** Data Modeling Working Group **Status:** Data COE Approved

Columns

Data Modeling Guideline #G23: Foreign keys should be named the same as the parent table's column(s) names. **Description:** Foreign keys should be named the same as the primary key of the table from which they are derived. Foreign key columns in a child table should not have additional words added to the column name to add meaning. **Rationale**: By using a similar name, it will be easier to associate it with the corresponding columns from which it was derived. Most modeling tools can be set to do this automatically. **Example:** If the parent primary key column is PARTY_ID, do not name the foreign surrogate key CUSTOMER_ID. In this case the foreign key should also be named PARTY_ID **Exception:** In the rare event that a parent has multiple relationships to the child, make the name meaningful. Such as table SYSTEM_INTEGRATION has a parent of SYSTEM twice because one is the "source" system and the other is the "target" system. There would be a need for two foreign keys on this table. Therefore, one could be named SOURCE_SYSTEM_ID while the other could be named TARGET_SYSTEM_ID. This follows the foreign key naming standard, but prefaces it with something meaningful to distinguish between the multiple foreign keys.	**Source:** Data Modeling Working Group **Status**: Data COE Approved

Constraints

Data Modeling Guideline #G24: Constraints on Primary and Foreign keys should incorporate "PK" or "FK" in the constraint name. **Description:** A meaningful name for the database constraints for a primary or foreign key should include an indicator as to which it is associated with. **Rationale**: By using an identifier in the constraint name, it will be easier to identify the usage of the constraint.	**Source:** Data Modeling Working Group **Status**: Data COE Approved

Indexes

Data Modeling Guideline #G25: Indexes shall have meaningful names using consistent patterns. **Description:** Physical database naming standards should be used. It is best if the index names are meaningful instead of randomly chosen and use a consistent pattern within the database. **Rationale**: Meaningful and consistent naming practices make the database objects easier to consume and maintain.	**Source:** Data Modeling Working Group **Status**: Data COE Approved

3.3 Definition Guidelines

Refer to ISO/IEC 11179-4 at http://standards.iso.org/ittf/PubliclyAvailableStandards/index.html **for guidance on Definitions of Entities and Attributes. Tables and columns should inherit the same definition guidelines.**

3.4 Data Model Management

Data Model Notation

Data Modeling Guideline #G26: Data Models should use IE notation. **Description:** IE, the most recognizable notation, should be used for data modeling notation. **Rationale**: IE is easily recognized by most modelers and business associates. A common notation in data models across the enterprise will have a common visual appearance and allow ease of readability. Modelers and other who are customers of data models will only need to learn one notation.	**Source:** Data Modeling Working Group **Status:** Data COE Approved

Data Model Versioning

Data Modeling Guideline #G27: A good data model management process is essential. **Description:** A comprehensive change management process must be in place to manage modifications to the model. Data Models need to be versioned as they are modified for new development and enhancement projects. A process needs to be in place to ensure backup of models occurs on a regular basis. The ability to revert back to a previous model is necessary. **Rationale**: Models are like any other piece of software being developed they needs to be versioned. Time and money will be wasted if project teams cannot rely on the integrity of the data model.	**Source:** Data Modeling Working Group **Status:** Data COE Approved

Data Model Subject Areas

Data Modeling Guideline #G28: Subject areas should be built from large models to target entities/tables based on a logical grouping. **Description:** A Subject area can be developed as a subset of a larger data model to target specific business functionality or to target key entities/tables for a given purpose. **Rationale:** By grouping common entities into subject areas the models are more organized for readability and reduce the number of entities on a visual page. Printing these small subject areas allows for a larger font for each entity thus more readable.	**Source:** Data Modeling Working Group **Status:** Data COE Approved

Data Modeling Guideline #G29: Subject areas should be named based on their business functionality. **Description:** To prevent misconceptions of the entire data model, properly label subject areas to reflect the purpose for which they were created. **Rationale:** Subject areas are based on business functionality and should be named accordingly to avoid confusion.	**Source:** Data Modeling Working Group **Status:** Data COE Approved

Data Model Presentation

Data Modeling Guideline #G30: Placement of entities in the data model should be done to make it easily readable. **Description:** On relational data models, position child entities under the parent entities or child entities above parent entities so the visual is either top down or bottom up. On dimensional models, position the facts in the center with the dimension tables surrounding the fact tables so the visual is center out. **Rationale:** By laying out the entities in their most appropriate places, it is easier to walk through and review the models. It is simpler to tell the story within the data model and clearly read the relationships from parent entity to child entity. Laying out the entities properly can quickly find errors and issues such as circular relationships.	**Source:** Data Modeling Working Group **Status:** Data COE Approved

Data Modeling Guideline #G31: Place relationships in the data model for readability without compromising the entity layout. **Description:** Minimize crossing lines. Avoid relationship lines that pass through entities. An entity should not be placed on a relationship line if it does not participate in that relationship. **Rationale:** By laying out the relationships to minimize crossing lines and eliminating relationships passing through entities that are not part of the relationship, the model is easier to read and more straightforward to tell the story within the structures. **Implementation:** Give the entity layout priority over the relationship layout. For example, if the only way not to have two relationships cross is to move the child entity above the parent entity, or move the parent entity toward the center and the child entity toward the border, keep the crossing lines instead of compromising the positions of the entities. Most of the time, slightly moving the entities or relationships improves the layout of the relationship without compromising the entity layout. Another option is to resize the entities will maintain the entity layout but allow for not crossing of lines.	**Source:** Data Modeling Working Group **Status:** Data COE Approved

3.5 Data Model Peer Reviews

Data Modeling Guideline #G32: A peer review of the logical data model should be conducted before creating the physical data model. **Description:** As part of the logical data model design on major changes and new development and before the physical data model is created, a data model walkthrough and review should occur between the data modeler and modeling peers. This should serve to evaluate the model on how well it captures requirements, the completeness of the model, how well the model leverages Reference Model structures, naming standards, readability, consistency, and quality of definitions. **Rationale**: Since the logical data model is the foundation for the physical data base design, it is important to interact with others that may have a broader or more global understanding and appreciation of the data beyond the scope of the project. A data model review by peers helps uncover some of the common mistakes made. The review provides the opportunity to collect suggested additions where content has not yet been created.	**Submitted by:** P&C Data Warehouse Data Modeling Standards and Guidelines **Status:** Data COE Approved

4. Tips

Data Model Presentation Options

Depending on the Data Modeling tool being used, an option to highlight areas of interest through presentation may be available to a complex model easier to read. When these techniques are used, it is important to remember that a legend must be included on the data model to ensure the audience understands the intent. This tip is neither a standard or guideline, but provided as an alternative when building a data model where presentation may make it easier to consume.

Font – The use of different fonts can be used to emphasis points of interest. For example, table names might be in a larger font, or a bold font. Columns that the modeler wants to draw attention to could be set to italics.

Color - Adding color to data model objects can provide a method of presenting a complex model in an orderly fashion that makes it easier to understand the groupings of data and/or relationships. When adding color, limit the colors so that a person who is color blind or those who may not have access to a color printer are not limited by the colors. Examples of how a data model could be color coded include inputs and outputs of the data, subject areas, type of table, such as a look-up table or views, or to distinguish dimension tables from fact tables or those tables not implemented as part of the scope of a given project.

5. Document Information

Exceptions

Standards are defined as rules that must be followed. If unique project requirements dictate deviations from the standard, the deviation should be documented in an Architect Decision Document for the project.

If multiple exceptions are granted over time for a given standard, the Data COE should be contacted for possible change to the Standard.

Document History

Version	Date	Author	Reviewer	Purpose	Status
0.02	8/7/2008	Data COE Data Modeling Working Group	Data COE Data Modeling Working Group	Review	Shared
0.03	8/18/2008				
0.04	8/22/2008				
0.05	8/28/2008				
0.06	9/15/2008				
0.07	9/23/2008				
0.08	9/29/2008				
0.09	10/14/2008				
0.10	11/13/2008				
0.20	11/17/2008	Chris Burns	Data COE	Approved	Shared
Enterprise Version 1.00	11/20/2008				Published
1.01	02/26/2014	Linda Macklin	Data Modeling User Group		Shared

Approvals and Contributors

Approvals
The following individuals have reviewed, contributed and approved the contents of this document:

Name	Role
Linda Macklin	Data Modeling Working Group Team Member
Felix Dashevsky	Data Modeling Working Group Team Member
Padmini Narayan	Data Modeling Working Group Team Member
Debby Cada	Enterprise Data Governance Advocate
Gauri Dandekar	Data Modeling Working Group Team Member
	PCIT Reviewer
Data COE	Data COE Reviewer & Approver

Standards Currency: location, accessibility, revisions

This document is located on the Nationwide IT Architecture knowledge repository. See the Data Modeling Standards and Guidelines folder for the latest version.

Comments