A Tutorial on QA of a Data Model

Barry Williams
www.DatabaseAnswers.org

A Tutorial on Quality Assurance of a Data Model

A. List of Activities

This Chapter describes the background to each Activity.
The next Chapter shows an example of how this works in practice.

1. Create a Top-Level Business Data Model

1.1 Types of Data Models

All the Data Models that we will be discussing can be described as Entity-Relationship Diagrams, or 'ERDs'.
They all show Relationships between Entities or Tables.
At the conceptual level, the 'Things of Interest', such as 'Organizations', are called Entities and at the Logical or Physical level they are called Tables, because they often appear as Tables in Databases.
At the Physical level, Tables are given names in the plural, such as Organizations, whereas at the Conceptual level they often appear in the singular, that is 'Organization'.
At the Logical level they might be either singular or plural.

A Top-Level Business Data Model can be created using Microsoft Word and is intended for business users and a non-technical audience.

The other Models referred to in this document will always be created by a Data Modelling Tool such as ERWin or IBM's Rational Rose.

They could be described as Conceptual, Logical or Physical Models.
Conceptual Models show the 'Things of Interest' which are in Scope, for example, Organizations and Products.
They may or may not include Keys and will certainly not include physical Data Types, such as the length of character strings.

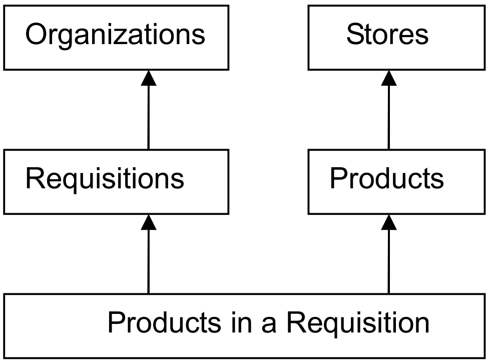
Logical Models will include Primary and Foreign Keys and often the Modelling Tool will provide a facility to generate a Physical Model form a Logical one.

Physical Models are often close to the actual design of an operational Database.
They will always show data types and field lengths.

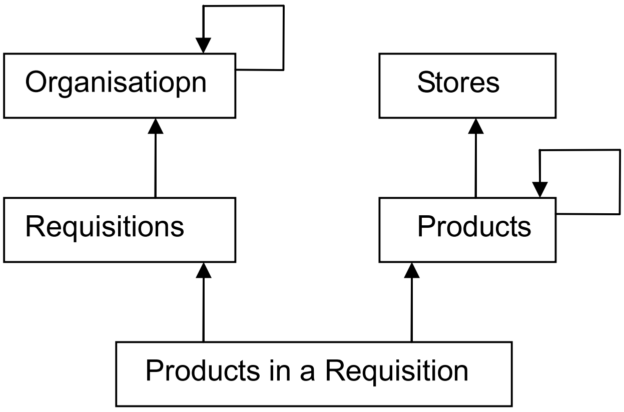
The Models shown in this document were created by Dezign, produced by a Dutch company called Datanamic.

1.2 Example of a simple Business Data Model

This Model was created in Word and shows Organizations, Requisitions and Products.
The flow of logic in a Data Models should go from top-left to bottom-right.
This means that the more fundamental Things are on the top and to the left.
This diagram is a good example.



This version shows that Organizations and Products each have a hierarchy so that an Organization is part of a higher Organization.
Similarly, a Product can be part of a more complex Product.
Which of these two you choose to use will depend on the audience.
In general, it is better to choose the simple option.



2. Draft the Business Rules

Business Rules are valuable because they define in plain English with business terminology the underlying relationships between the Terms that appear in a Data Model.
The User community will then be able to agree and signoff the Rules.
Here is a small example.

Nr	TABLE	DESCRIPTION
D.1	Requisitions and Organizations	A Requisition must be raised by a valid Organization. Not every Organization will raise a Requisition. Therefore the Relationship must be one-to-many with a mandatory

		condition at the Organization end and an optional condition at the Requisition end.
D.2	Requisitions and Products	A Requisition must refer to valid Products. Therefore the Relationship must be one-to-many with a mandatory condition at the Product end and an optional condition at the Requisition end, because not every Product will appear in a Requisition.
P.1	Products and Stores	Products are kept in Stores

3. Draft a Glossary of Terms

It is very important to establish agreed definitions of terms and words in common use.
This is a small example.

TERM	DESCRIPTION	COMMENT
Requisition	A Request or Requisition for Products to be supplied to the Requesting Organization.	
Product	An Asset that can be separately ordered. It can be a Component and a part of a larger Assembly. It can be very small, such as a Washer, or very large, such as a Tornado aircraft.	Very large Products can be subject to a separate Requisition Process.

4. Check that the Data Model is correct

There may be errors which have a simple explanation.
For example, the incorrect use of the Modelling Tool.
Any errors should be discussed and resolved with the Modeller and the Users.

This is where the Glossary and Business Rules are very valuable.

5. Review with Users

At this point, review the Business Rules and the Glossary with Users and aim to get Sign-Off.
Make any necessary changes to format and contents.

6. Check Normalised Design

6.1 Normalised Design

This discussion applies to Entity-Relationship Diagrams (ERDs) and not to Data Warehouses.
We will start by defining the Rules for Normalisation so that we can recognise cases where they have been broken.

Rules for Normalisation
Rule 1 :
A little background is appropriate at this point.
The theory which provides the foundation for Data Models and ERDs, was developed in 1970 by an Englishman called Ted Codd, who was a research scientist with IBM in California at the time.
One of his rules can be summarised as :-
“The Data in a Table must belong to the Key, the Whole Key and Nothing but the Key, so help me Codd”.

This means, for example, that a record in a ORGANIZATIONS Table must contain data only about the Organization, and nothing about people in the Organization, or activities of the Organization.
It might include things like the name of the Organization and when the Organization was founded.
Check 1 : Can the values of every data item in a table be derived only from the Primary Key ?

Rule 2 :
Another of Codd’s rules stated that derived data must not be included.

For example, the headcount for an Organization would not be included in the Organizations Table because it can be derived by counting the records of members in the Organization.
Check 2 : Can any data item be derived from other items ?

Rule 3 :
There must be no repeating groups in a Table.
The one uncomfortable exception is that Addresses.
They are very often stored as a number of repeated lines called 'Address_Line_1', 'Address_Line_2', and so on.
Check 3 : Do any column names repeat in the same table?

Rule 4 :
An item of data must only be in one Table.
For example, the name of a Organization would appear only in the Organizations Table.
Check 4 : Does the same item of data in appear in more than one table ?

6.2 Reference Data

6.2.1 Background

A list should be made of the Reference Data referred to in a Data Model.
When the list is complete it should be analysed for consistency.
For example, there will not usually be any relationships between the Reference Data.
However, if there are any, then they should be sensible and consistent.
For example, a Town might be in a County which would be in a Country.
These could all be classified a Reference Data which has relationships which should be validated.

Typical Reference Data could include Job Titles, and Types of Products.
In passing, we should note that Organizations, Job Titles and Products are all examples of hierarchical structures.
Job Titles will change only very, very rarely.
However, when they are stored in a Table which is joined to itself then, of course, the Table will havea Recursive Relationship to itself.
Therefore, wherever these occur, we would expect to find compact Data Models that include a great deal with compact and powerful structures.

6.2.2 Standards

Any appropriate national, international Standards must be considered when values for Reference Data are decided.
These include MOD, NATO and ISO standards.
For example, ISO publishes standards for Country Codes and NATO maintains standards for Product classification.
Therefore any Data Model relating to Products should consider this standard and where appropriate the necessary Tables should be added to the Model.

6.3 Slowly-Changing Data

The classic example of Reference Data which never changes is a Calendar.
The values are predictable for hundreds of years ahead.
There is a category in between which is usually called 'Slowly-Changing Data'.
This applies where the values of the data changes on roughly a six-monthly basis.

Data about Categories and Types is often fixed values but some can change infrequently.
For example, a new Aircraft Type was introduced with Unmanned Aircraft.
The values then became Fixed-Wing, Rotary and Unmanned.
This would be an example of Slowly-Changing Data.
This highlights the fact that what constitutes Reference Data can be subjective and may be defined differently in Data Models created by different people or organisations.

6.4 Check Normalisation

Check Nr	GOOD	OK (Y/N) ?	DESCRIPTION

1	Y		Can the values of every data item in a table be derived only from the Primary Key ?
2	N		Can any data item be derived from other items ?
3	N		Do any column names repeat in the same table?
4	N		Does the same item of data in appear in more than one table ?

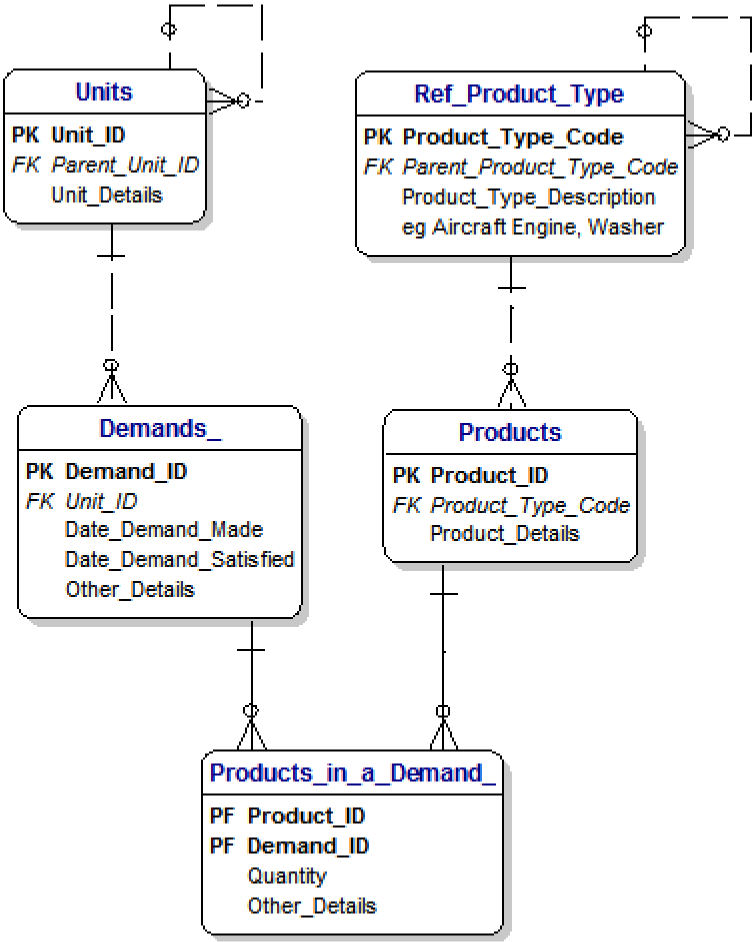
7. Look for Design Patterns

7.1 Some Examples

This Data Model shows examples of Design Patterns for One-to-Many and Many-to-Many Relationships, Reflexive Associations and Reference Data.

PK stands for ‘Primary Key’ and FK stands for ‘Foreign Key’.

PF, which is shown in the Products_in_a_Requisition Table, stands for Primary and Foreign Key’. This is a Primary Key in one Table which is also a link to another Table, where is also a Primary Key.



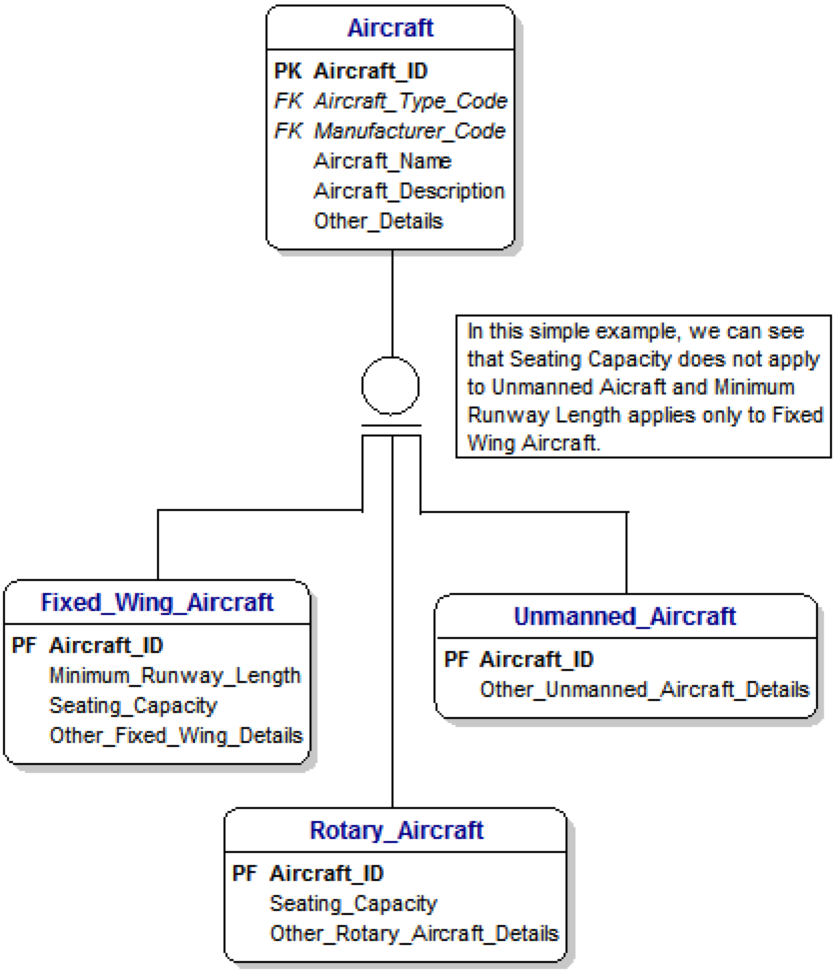
7.2 Inheritance

More details are provided in Chapter 2. Concepts in the document entitled “How to Understand a Data Model”.

We use the concept of ‘Inheritance’ where have Super-Types and Sub-Types.
Inheritance is exactly what it sounds like.
It means that at a high level, we identify the general name of the ‘Thing of Interest’ and the characteristics that all of these Things share.
For example, an Aircraft will have a name for the type of Aircraft, such as Tornado and it will be of a certain type, such as Fixed Wing or Rotary.

At the lower level of Fixed-Wing Aircraft, an Aircraft will have a minimum length for the runway that the Aircraft needs in order to take off.

This situation is shown in the following diagram :-



7.3 One-to-One Relationships

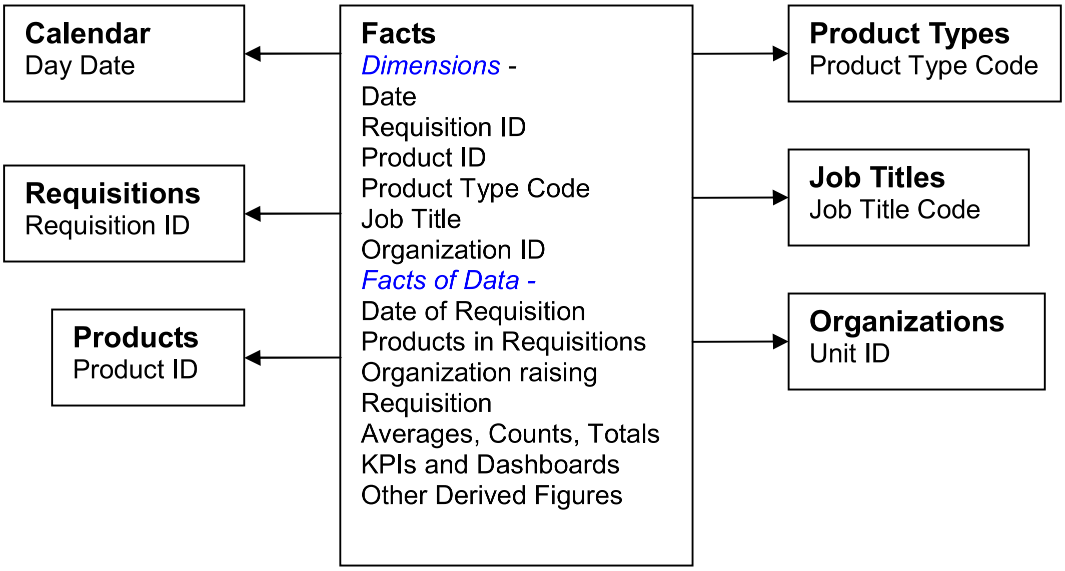
We can remind ourselves that Rule 1 above states :-
“The Data in a Table must belong to the Key, the Whole Key and Nothing but the Key, so help me Codd”.
One implication is that there should not be a One-to-One Relationship between two Tables in a Model because the data can be combined into one Table with the same Primary Key.

However, there is an exception to this which is when a one-off event can occur which involves a substantial amount of data.
In that case, it would not be good to create a large number of fields which will be blank in the large majority of cases.

For example, when a Soldier joins the Army there might be data that is involved only with the joining details.
The basic data for the Soldier will be part of his or her basic records – such as Date of Birth and Place of Birth.
If a separate Table exists for ‘Joining Details’ then it would contain such things as date and place of joining.
Then the Soldiers Table would have a One-to-One relationship with the Joining Details Table.
In other words, it can sometimes be acceptable to see a One-to-One in a Data Model.
If that happens, it is necessary to establish the associated Business Rules to clarify the conditions.

8. Review any Data Warehouses

This Section is relevant if the Data Model includes a Data Warehouse or Data Mart.
A Data Warehouse can be a Star or a Snowflake Design.
This diagram shows a typical Data Warehouse.
It is a Star structure with only one dimension for the related Dimension Tables.
The arrows point from Children to Parents.



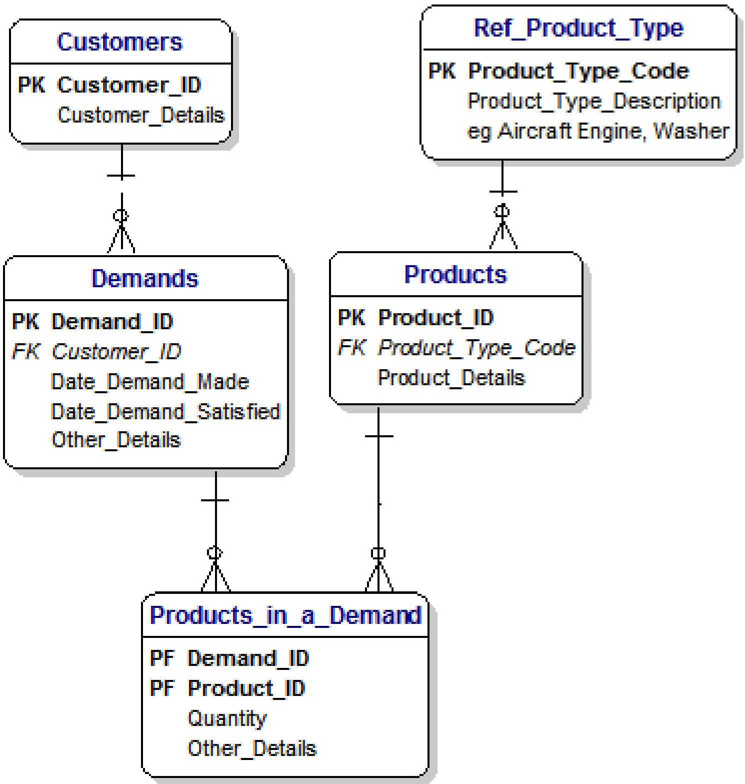
9. Check Naming Standards

At this Step, we check for compliance with Naming Standards.

For example, a typical standard might state that Field names should be specified with underscores linking related words and first letters in capitals, such as Organization_ID.

In the absence of any explicit Standard, this should be the default.
This is shown in the Model in Section 7.1 and also in this one.

We might say that Naming Standards are ‘nice to have’.
In other words, they are not essential but they reflect Best Practice.



10. Check for Consistent Data Types

It is important to check for consistent Data Types and Lengths for two reasons :-

1. it avoids nasty surprises when a physical Database is generated from the Data Model
2. it is an indication of the professionalism of the manner in which the Data Model was produced, unless it has been reverse engineered from a Database, in which case these design considerations do not apply.

For example, names should always be the same, or should be handled in a way that handles any differences in a way that ensures consistency.
Typically, a longer name should be explicitly truncated to a shorter value where appropriate.

In the absence of any explicit Standard, the default for Names or Address Lines should be VARCHAR(255) or VARCHAR2(255) for Oracle.
For other character strings they should default to Memo or Text.

11. Check for Defaults

We would like to see Default Values used wherever possible because they increase the discipline enforced by the Model and they indicate that a thorough analysis was carried out during the creation of the Data Model.
For example, a ‘Start Date’ could default to the current day, or the ‘System Date’.

12. Determine the Assurance Level

This could be either :-

i. Acceptable
ii. Acceptable with Reservations
iii. Not Acceptable

B. Checklist for Quality Assurance

This Checklist is based on the concept of the Data Model Scorecard which was originated by Steve Hoberman.
It is discussed in Steve's latest book, entitled 'Data Modeling for the Business'.

Nr	FEATURE	IMPORTANCE	Yes/No	COMMENT
1	Can the Scope of the Model be defined ?	Essential		There must be clear alignment of the Model with the business and the User community.
2	Can the Requirements be defined ?	Essential		Requirements must be defined.
3	Does the Model meet the Requirements ?	Essential		The Model must meet the Requirements.
4	Is there a comprehensive Glossary of Terms ?	Essential		Very important that the value of a Glossary is recognised
5	Have comprehensive Business Rules been defined ?	Essential		Very important that the value of Rules is recognised
6	Normalisation Checks (Good value of Y or N) 1. Can the values of every data item in a table be derived only from the Primary Key ? (Y) 2. Can any data item be derived from other items ? (N) 3. Do any column names repeat in the same table? (N) 4. Does the same item of data appear in more than one table ? (N)	Desirable		The Model might be OK even if it fails all of these Checks. The power and flexibility of the Relational Approach makes it possible to handle all sort of errors and still provide the foundation for an operational Database.
7	Is there compliance with any relevant Data Standards ?	Desirable		Not essential
8	Does the Model use relevant Design Patterns ?	Desirable		Might be OK even if Design Patterns can't be identified. But if it isn't laid out well it brings into question the professionalism of the creators of the Model.
9	Is the Model easy to read and understand ?	Desirable		The flow of logic in a Data Models should go from top-left to bottom-right. It might be OK even if it can't be read and understood. But if it isn't laid out well it brings into question the professionalism of the creators of the Model.
12	Can any repeating groups be identified ?	Not critical		Not critical because the Database will work OK. Usually indicates denormalisation for performance reasons. In other words, this can be acceptable for a Database design. For example, Address Lines often repeat.
12	Can any derived data be identified ?	Not critical		Not critical because the Database will work OK.
13	Does the Model follow Naming Standards ?	Not critical		Cosmetic but is a measure of the professionalism
---	---	TOTAL SCORE		

If the answers to all the Essential features is Yes then the Model is Acceptable.

If any of the ‘Essential’ Questions have a No answer, which the Model is not Acceptable.
Any No answers to ‘Desirable’ or ‘Not critical’ Questions do not affect the acceptability of the Model but mean that it could be improved.

RESULT	RATING	COMMENT
All Essential Features are Yes	Acceptable	
Any Essential Features are No	Not Acceptable	The Essential items are top priority for improvement.

Typical Summary

The results of a of a typical Model might result in this summary :-
“Reservations are that the documentation does not demonstrate that the Data Model meets the User Requirements.
The Data Model shows some weaknesses which the supplier has agreed to address.”

Follow-Up Remedial Action

A reasonable result of a QA analysis would be the identification of some problems that could be rectified fairly easily and quickly.
This applies to things like documentation and naming standards.
The appropriate Remedial Action will depend on the context and scope of the Data Model.

For a Health Check :-
No action is required beyond the presentation of a Report because the QA is simply to establish the ‘As-Is’ situation.

For a proposed Application :-
It is essential that the Model accurately meets the User Requirements.
If it does not, then it must be corrected in discussion with the Users and the Modeller.

For Data Migration :-
It is essential that the Model is correct at the detailed level of Tables, Fields and Data Types.

C. A Case Study

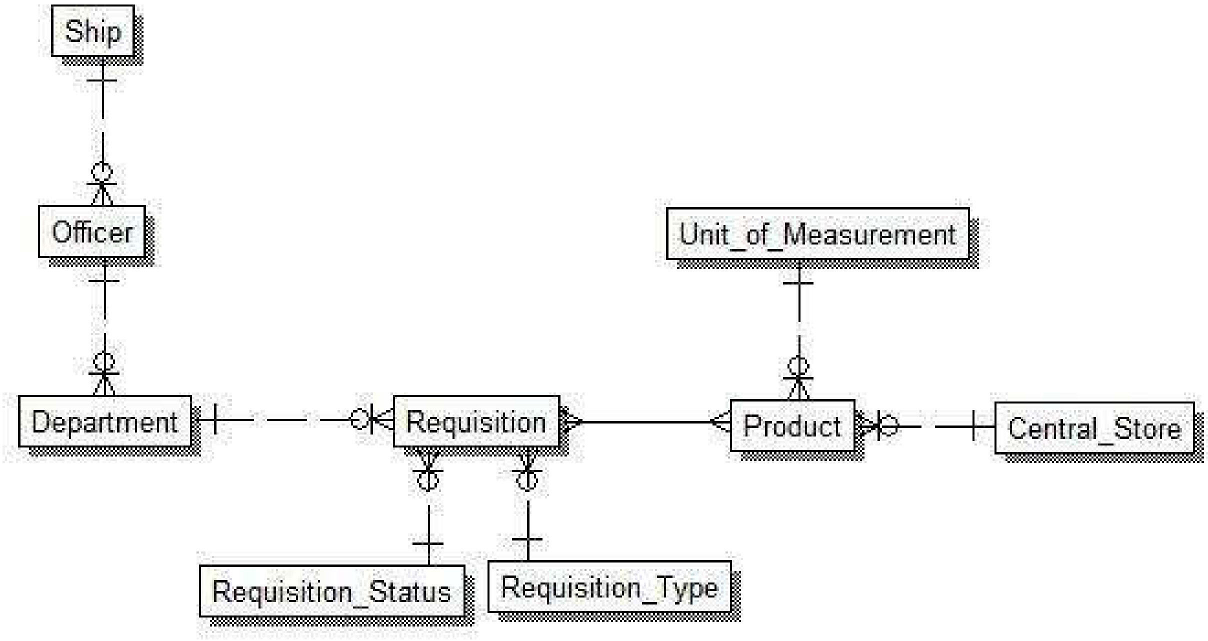
This Case Study provides an example of the Tutorial in action.
It includes blank Templates and sample Templates which are guidelines.

Step 1. Create a Top-Level Business Data Model

1.1 Background

Let's assume that a Data Model has been provided by a Third-Party, such as a Delivery Partner.
The first Step is to understand the Data Model by creating a Top-Level Business Data Model.

Here is the Data Model from the Delivery Partner that we will use as an example :-



1.2 Our Conclusions

Our conclusions are that this is not a good Data Model.
Reasons include :-

- It contains Reference Data which is not appropriate at the Top Level
- There is no description of the functional area that the Model supports.

Our first activity therefore is to produce an equivalent Business Model that we like that we can use as the basis for discussion.

Corrective Actions include :-

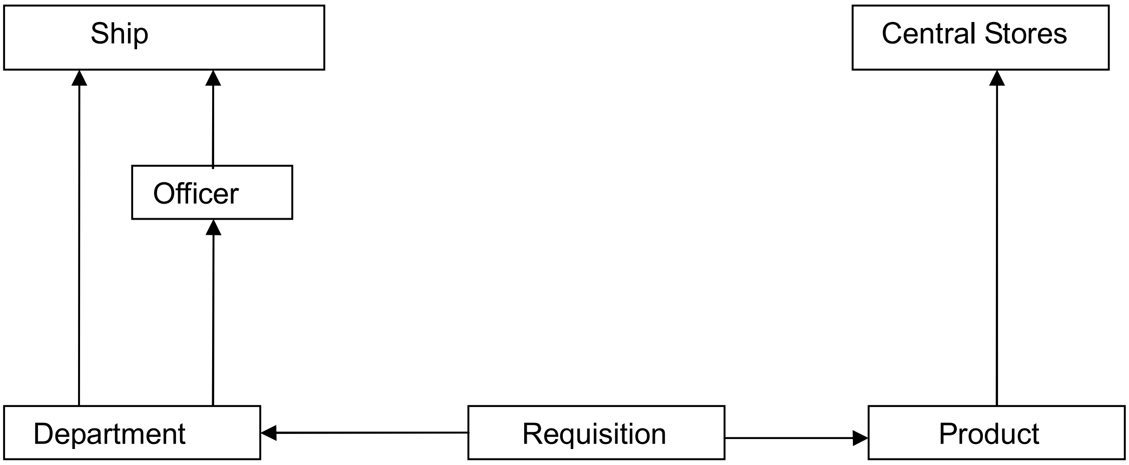
1. Create a simple Business Data Model
   This should be a Model in Word that does not include Reference Data
2. Produce a short description
3. Create a Glossary of Terms
4. Define the representative Business Rules
5. Identify the intended Users and the Owners of the Model

1.3 Functional Description

In this diagram, arrows point from Children to Parents.
The Scope of the Data Model includes Requisitions for Products from Organizations.
The Functional Description is a simple one-liner “Organizations raise Requisitions for Products”.

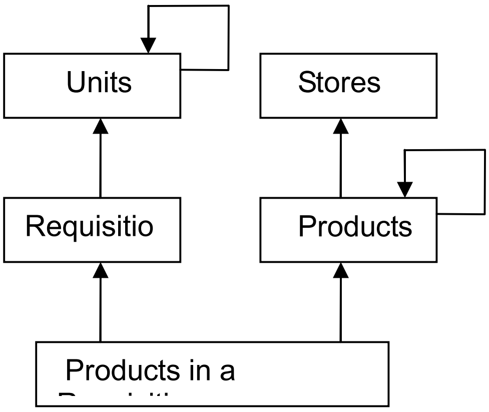
1.4 A Specific Model in Word

The Specific Version is consistent with the Generic Version and looks like this.



1.5 A Generic Model in Word

In this diagram, arrows point from Children to Parents.
Our Generic Data Model (from Section 1.2) looks like this :-

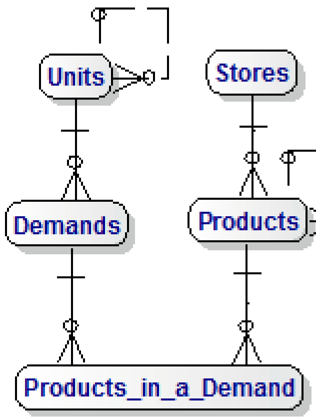


1.6 A Top-Level Generic Data Model

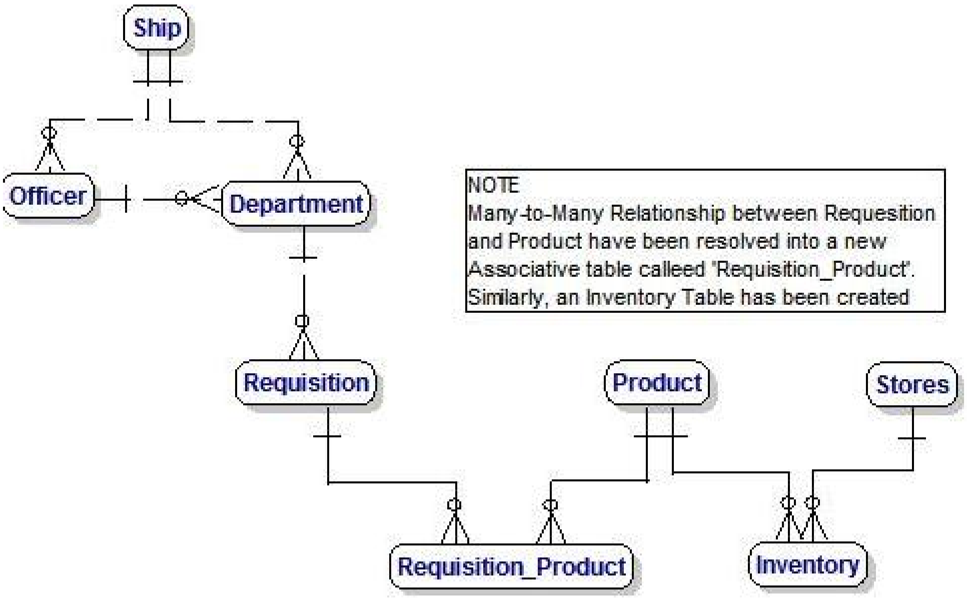
This is a top-level Model that was created using a Data Modelling Tool.
It shows useful detail, such as details of the Relationships.
It also replaced a Many-to-Many with two One-to-Many Relationships.

This diagram is a generic version of the one below and is useful to help in providing a higher level context for lower-level, more specific Models.

An additional level of detail shows the ‘Rabbits Ear’s relationship that implements hierarchical relationships for Organizations and for Products.



This Model corrects an error in the original Model that we were given.
The error was that a Ship has Departments independent of Officers.
This diagram shows that Relationship correctly.



NOTE
Many-to-Many Relationship between Requesition and Product have been resolved into a new Associative table calleed 'Requisition_Product'.
Similarly, an Inventory Table has been created

Step 2. Draft the Business Rules

These Rules must be phrased in unambiguous English.
Where possible, the English should make it possible to implement a Rule in a Data Model.
For example, Rule 1 makes it clear that there is a One-to-Many Relationship between a Ship and an Officer.

1. A Ship is staffed with many Officers.
2. Ship's Departments raise Requisitions.
3. A Requisition must be authorised by an Officer.
4. An Officer is assigned to one Ship at any point in time.
5. An Officer is assigned to one or many Ship during the course of their career.

Templates

Here is a sample Template for Business Rules :-

Nr	RELATES TO	OWNER	DEFINITION
BR.D.1	Customers, Requisitions	John Doe	A Customer can raise zero, one or many Requisitions.
BR.D.2	Customers, Requisitions	TBD	A Requisition must be associated with a valid Customer.
BR.D.3	Requisitions, Products	John Doe	A Requisition can refer to one or many Products.
BR.D.4	Requisitions, Products	John Doe	A Product can appear in zero, one or many Requisitions. Therefore, there is a Many-to-Many Relationship between Requisitions and Products.

Step 3. Draft a Glossary of Terms

This is a sample Template.

TERM	AUTHOR	DEFINITION
Customer	John Doe	Any Organization that can raise a Requisition
Requisition	John Doe	A request for Assets to be supplied. The format of a request can be an electronic message, a paper Form and so on.
Product		An Asset that can be supplied on request. It can be something small, like a Pencil, or something large, like an Printer. The term Equipment is reserved for major items, such as Printers. The word Asset is used to refer to smaller items, such as Pencils

Step 4. Check that the Data Model is correct

The Rules will help in determining whether the Model is correct.
In this case, there is an error in that Ships are shown coming in between Ships and Departments.
The reality is that Departments exists without Officers.
This is corrected in the Top-Level Data Model in Section 1.6..

Step 5. Review with Users

Review and revise as necessary.

Step 6. Check Normalised Design

The design looks normalised and therefore is acceptable.

The Reference Data looks appropriate and is not related and therefore is acceptable.

Step 7. Look for Design Patterns

This Business Model shows these examples of Design Patterns :-

- a One-to-Many Relationship between Ship and Office
- a Many-to-Many Relationship between Requisition and Product

It does not show Inheritance but in general we would not expect to find it.
There are a number of reasons why Inheritance does not appear.
For example :-

- Inheritance is not appropriate in this case
- Inheritance does not show in a Data Model for a physical Database.

Step 8. Review any Data Warehouses

In this Case Study, this Step is not necessary because we do not have a Data Warehouse or Data Mart.

Step 9. Check Naming Standards

Standards that are common include ;-

1. Initial Capitals with lower case elsewhere – for example, Organization_ID
2. All capitals – for example, ORGANIZATION_ID
3. Lower case everywhere – for example Organization_id

Any of these Standards is acceptable.
If no Standard has been established, then Number 1 is recommended.

Step 10. Check for consistent Data Types

This Check requires a Physical Data Model or some other document that includes this level of detail.
The procedure then is to visually scan the documents or use the Domain feature of the Modelling Tool or perhaps SQL to look for discrepancies.
The Domain feature allows you to define standard data types for any Data Item that occurs frequently and then use this Domain for every occurrence of the Data Item.
For example, a Name could be defined as a Variable-Length Character string with a length of 255.
Then whenever a Name appears in a Model, the Modeller can select this Domain as a convenient shorthand and also a simple way to enforce consistency.

It will be necessary to analyse any discrepancies and decide on a standard to resolve them.

Step 11. Check for Defaults

Default values are a powerful technique for adding values in a Data Model.
They can be used enforce consistency.
Probably the most common example is to specify that the date of entry and creation of a new record should be the current System Date.
This applies to new Customers, Orders and the Date of any Payment or Adjustment and so on.

Step 12. Determine the Assurance Level

Appendix A defines the process to be followed and discussed appropriate remedial follow-up action.