LOGICAL DATA TYPES

During the database development process, as we move from Chen's conceptual model to a logical model, we want to use Codd's Relational Data Model (RDM) to guide us to a logical data model that conforms to the RDM.

The Chen's definition of an entity is the detailed description of the entity. This is very similar to a 'metes and bounds' title of a plot of land. This definition of the entity (application domain data) will contain the list of all of the entity's attributes, integrity constraints, data value restrictions, etc.

In the transition from the conceptual model to a logical data model,

- the RDM relation labels become the logical entity names
- the RDM domain labels become the logical entity's attribute names

Although Codd is specific about the relations (entities) that make up the logical data model and the domains (attributes) that make up the relations, he does not define the domains other than to say that a domain is single valued and it is not a relation.

To complete our logical data model we need to be able to define the logical attributes. Since Codd does not specify how the domains are defined, we will use the following data to define an attribute

- the domain name becomes the attribute name
- the attribute will have a definition or description
- the attribute will have a logical data type

We know that the logical data types are not the data types used in the DBMS implementation of the RDM since these data types are the physical data types, strings of bits that make up text and numbers. The only logical data type implemented in a DBMS is datetime which allows us to record and manipulate the clock and the calendar as defined by the BIPM. Other organizations have defined other logical data types, for example, the UPU has defined postal address, the ITU has defined telephone number, the IUGG has defined geographic location (latitude, longitude and altitude), etc.

In the logical data modeling process, the questions are:

1. what are the logical data types (Gotlieb and Gotlieb),
2. what is the definition of a particular logical type
3. how can the logical data types be transformed into physical data types

In the pages below, we will address each of these questions in turn.

To find the answer to question 1, we partition the entity attributes into two levels of non-overlapping subcategories.

At the first level, the data describing an entity attribute is either a measurement or a description of the entity.

At the second level, the measurement can be a measure of

- money
- a physical property such as height, length, etc.
- a geographic location
- discrete time such as a date, time or event
- continuous time

As the diagram shows similar decompositions can be carried for descriptions. In essence, we have taken a generalized entity attribute and decomposed it into non-overlapping data types as in Figure 1.

Of course, different application domains would produce different list of logical data types.

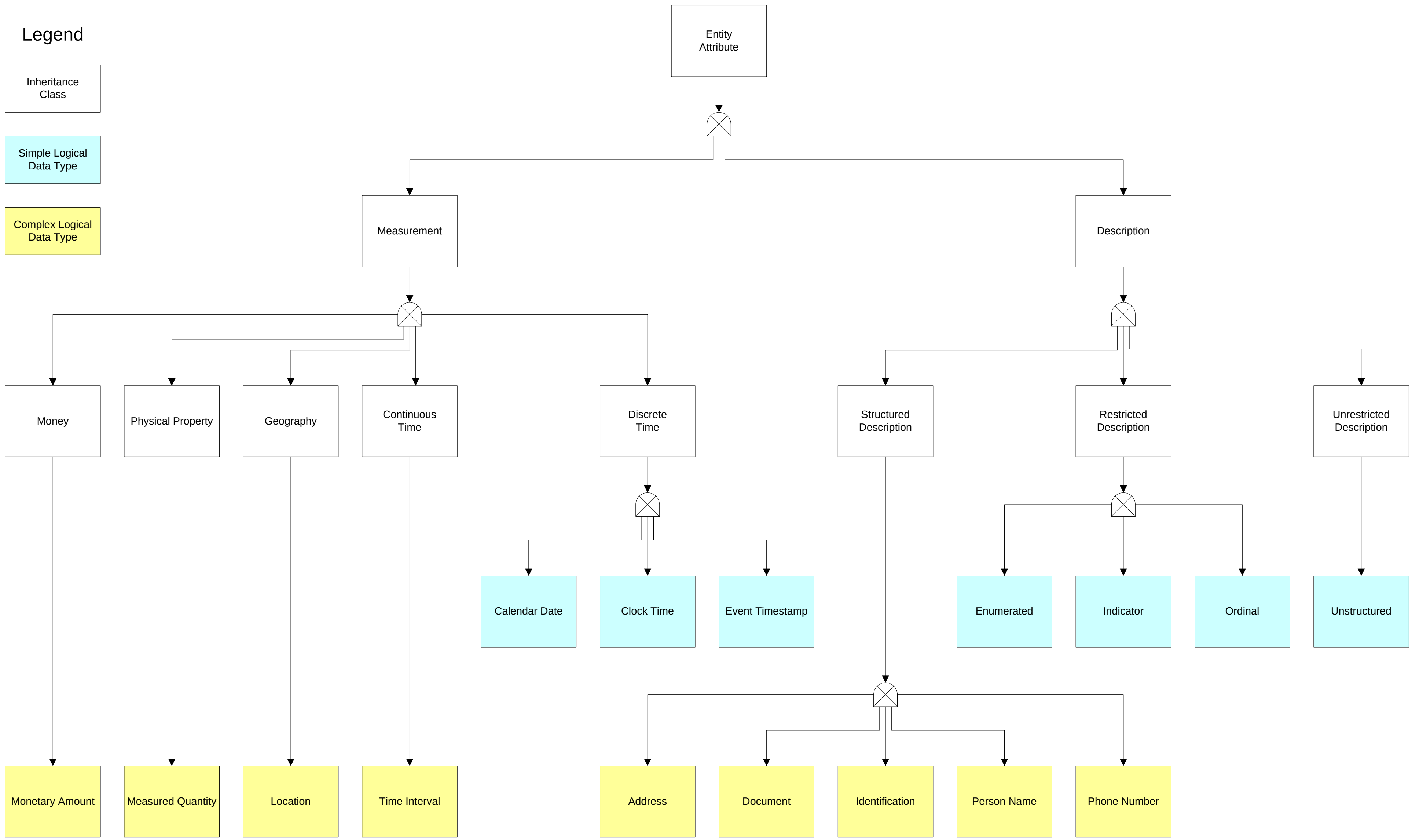

Figure 1. Logical Data Type Structure Chart

To answer question 2, the definitions of the data types are given in Table 1, Although there can be many complex structured data type that are defined by the business.

Logical Data Type Name	Description	Inheritance Class	Data Type	Representation	Default Values
monetary amount	The amount of money. An attribute that represents an amount of money in a business transaction.	Money	complex	value currency unit of measure	zero local currency total
measured quantity	A physical measurement on an entity. An attribute that represents a physical measurement on the business data object.	Physical Property	complex	value unit of measure	zero default
location	A unique location in the world. There are several possible representations: measurements of latitude, longitude and altitude; a physical address provided by a government; etc.	Geography	complex	depends on the representation chosen by the business	
time interval	An interval of time between a start date and an end date. Measured in days.	Continuous Time	complex	start date end date	beginning of time end of time
calendar date	A date in the Gregorian calendar. This logical data type is implemented in the DBMS	Discrete Time	simple	single date	beginning of time or today or end of time
clock time	The time of day within a date. This logical data type is implemented in the DBMS	Discrete Time	simple	single time	12:00:00 AM
event timestamp	The date and time of a recorded event, including year, month, day, hour, minutes, seconds, etc. This logical data type is implemented in the DBMS	Discrete Time	simple	date time	today, 00:00.000

Table 1a. Measurement Logical Data Type Definitions

Logical Data Type Name	Description	Inheritance Class	Data Type	Representation	Default Values
address	The postal address where mail can be delivered.	Structured	complex	Defined by the UPU	
document	The identification of a line in a physical document	Structured	complex	depends on the representation chosen by the business	
identification	The identification of an internal entity (employee, customer, supplier, etc.) by an external agent (country, IRS, SSA, etc.)	Structured	complex	depends on the representation chosen by the business	
person name	The name of a person	Structured	complex	depends on the representation chosen by the business	
phone number	The phone number (logical address on the telephone network)	Structured	complex	Defined by the ITU	
enumerated	A data type that is constrained to a list of allowed values. For example: gender, color, unit of measure, currency, etc. The list is resolved as a lookup table. The list should by a description of the entity and not a classification.	Restricted Description	simple	single coded value	zero
indicator	The allowed answers to a yes/no question. The values can only be yes, no or unknown.	Restricted Description	simple	single coded value	unknown
ordinal	The order or numeric sequence of a set of data occurrences	Restricted Description	simple	single numeric value	zero
unstructured	a string of length 255	Unrestricted Description	simple		NULL

Table 1b. Descriptive Logical Data Type Definitions

Database Engineering addresses the problems posed in question 3. The simple logical data types can be easily implemented as a single DBMS data type with straight forward operations. The complex logical data types need to be implemented as structures (multiple DBMS columns and data types) with complex operations required when manipulating and comparing the structures.

REFERENCES:

Codd, Edgar, “A relational model for large shared databanks”, Communications of the ACM, Vol. 13, No. 6, Jun 1970

Chen, P., "The Entity-Relationship Model-Toward a Unified View of Data"; ACM Transactions on Database Systems, Vol. 1, No. 1, pp. 9-36, Mar 1976

Gotlieb, C.C. and Gotlieb, Leo R., Data Types and Structures, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1978