# FROM A RELATIONAL TO A MULTI-DIMENSIONAL DATA BASE

**David C. Hay**
**Essential Strategies, Inc**

In the buzzword sweepstakes of 1997, the clear winner has to be "Data Warehouse". A host of technologies and techniques has promised finally to achieve for the business world a dream that has been around for nigh on thirty years — the ability to obtain data about an enterprise directly from the computer in a form that is appropriate to the particular question being asked.

In 1970, your author developed a system which allowed a user to get reports on sales and sales targets, for product groups, parts of the country, and various time periods. If a particular line looked suspicious, it was possible to select that line (which, for example, might have described a product group in a region) and get a report showing it broken out by product, or by city. Now understand that the user interface was a ten-character-per-second teletype, and the whole thing was written in Basic. It wasn't as "user-friendly" as it might have been, and we've learned a lot about data organization since then, but the underlying objective hasn't changed.

## Three schema Architecture

At about the time your author was developing that data warehouse, the "three-schema architecture" was being defined. It portrayed the world of data management from three (and ultimately four) perspectives. See Figure 1.

On the left you see the set of views of data that are held by various people in an organization. Each "external schema" is slightly different, reflecting the particular requirements of that person's job. Originally, computer systems developed for such people reflected only those views.

On the right you see the way data are handled by the computer. This "internal schema" is itself in two parts. The "logical schema" reflects the way data are organized by a particular data base management system. In the Oracle world, this is in terms of tables and columns. The "physical schema" describes the way data are actually stored on a particular device, in terms of tablespaces or cylinders and sectors.

In the middle you see the "conceptual schema" — the inherent, underlying structure of the data. This is the composite of all the external views. Where different external views might define a product differently, the conceptual view contains a definition from which all others can be derived. The insight offered by the three-schema architecture was that each of these schemata can be addressed independently. It is possible to understand the conceptual structure of data without having to think about how the they will be stored on the computer at the same time. It is possible also to separate this conceptual structure from various people's views of the data.

The theory behind relational technology was that you could build a data base along the lines of the conceptual schema, and then define views that would reflect the specific requirements of each user. One day computer technology will be powerful enough to make that a reality, and the whole business of data warehouses will disappear. For the present, however, processing realities make it necessary to create separate sets of tables for each external schema and copy data from the central data base into them.

The data warehouse phenomenon is recognition of the fact that we really do have to move data from the conceptual model into the external views. Slowly, tools are beginning to emerge that allow us to do that readily, and other tools are making it easier to get data from those external schemata.

## The Conceptual Schema — the ERD

The tool for representing conceptual data has long been the entity/relationship model, otherwise known as the ERD or simply, the data model. It can be used as a tool for representing any data structure, anywhere in the three-schema architecture, but its significance lies in its power to describe and present the conceptual model.

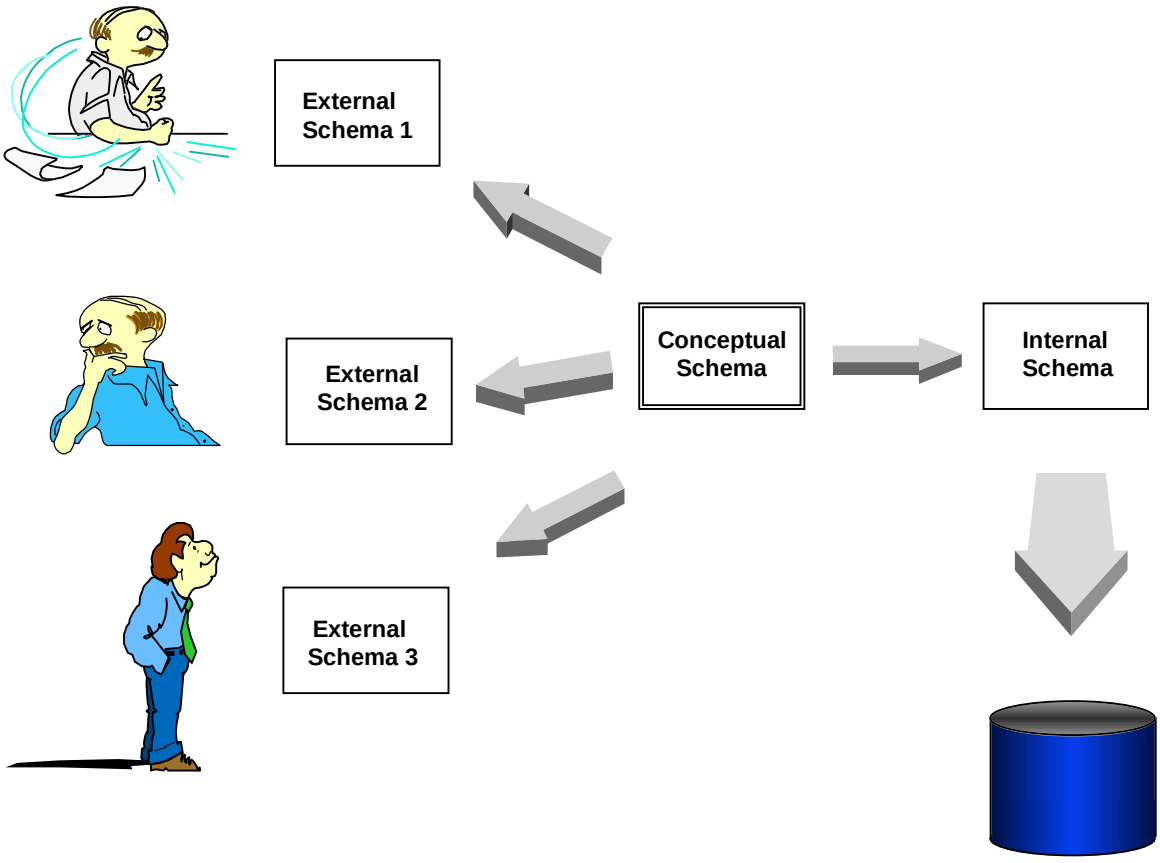



***Figure 1: Three-Schema Architecture***

The priority in modeling the conceptual schema is to represent the organization's essential structure. An entity is a thing of significance to the business, about which it wants to hold information. It exists independently of any particular individual's view of it. The relationships between entities represent relationships between the things in the business that are being portrayed. The model's scope is the complete set of things that any of the end users might want information about.

An important aspect of the conceptual model is that it is "normalized". That is, its structure follows specific rules to ensure that there is no redundancy. Every data element is identified as to where it belongs, and it is placed only there. What particular thing does it inherently describe, however many other things it might be related to?

An ERD implemented as a corporate data base will be reasonably stable, since its structure reflects the

inherent structure of the business. While the activities of the enterprise may change, this structure will not very much. Such a database is a reasonable basis for operational systems, since these produce transactions one at a time, and it is simple to define where each datum should go. For this reason, such a data base is sometimes called an "Operational Data Store" (ODS).

As an example, look at the portion of a corporate data model shown in Figure 2. In this we have PARTY, which may be either a PERSON or an ORGANIZATION. An ORGANIZATION in turn may be either an internal organization such as a department or division, or an external organization, such as a customer or a vendor. Each PARTY may be *subject to* a PARTY PLACEMENT *at* a SITE, where each SITE, in turn, must be *located in* a GEOGRAPHIC LOCATION. Each GEOGRAPHIC LOCATION may be *part of* one or more other GEOGRAPHIC LOCATIONS, and may be *composed of* one or more other GEOGRAPHIC LOCATIONS. This is represented by the entity LOCATION STRUCTURE, which represents each occurrence of one LOCATION being in another. Each GEOGRAPHIC LOCATION must be either a STATE a COUNTRY, or an OTHER LOCATION.

PEOPLE are shown to be working for ORGANIZATIONS, via the entity EMPLOYMENT, where each EMPLOYMENT must be *of* a PERSON *in* an ORGANIZATION. Each EMPLOYMENT, in turn, may be *the basis for* one or more POSITION ASSIGNMENTS, where each POSITION ASSIGNMENT must be *for work at* a SITE and *to* a POSITION. Each POSITION, in turn, must be *with* an ORGANIZATION. Each POSITION must also be *an example of* a POSITION TYPE.

We see also that PARTIES (both PEOPLE and ORGANIZATIONS) are related to each other via BUSINESS RELATIONSHIPS. For example, some ORGANIZATIONS are companies and some are departments. A BUSINESS RELATIONSHIP can represent the fact that a particular department is part of a particular company. Both EMPLOYMENT and BUSINESS RELATIONSHIP are placed in time via the attributes "Effective date" and "Until date".

Note the attempt here to keep the concepts represented as general as possible. There is no telling but what we may one day want to know about the employees of our competitors, or our customers, so we define employment for all organizations. For that matter, organization is defined in as general terms as possible. There is no telling but what we may one day discover that we want to keep track of government agencies, or labor unions, and we want to know that they will fit as well as those we know about now. Sites in this model may be anything — offices, warehouse locations, or production lines, to name a few. The geography may also be anything — states, counties, standard statistical metropolitan areas, or whatever.

All these things can be accommodated by this model.

## The External Schemata — Data Marts

While data are collected one at a time, they are usually retrieved in groups. Since the average business is complex, the average data model (and its resulting corporate data base) tends to be complex as well. To obtain a piece of information derived from several columns might require extensive navigation of this data base, involving many tables. While the conceptual structure may be the most efficient as a way of storing data, it is not necessarily the best structure for retrieving them.

A data warehouse, therefore, consists of not only the operational data store (reflecting the conceptual schema) but also "data marts" reflecting the external schemata. Each data mart aggregates data in terms of the particular expectations of what someone will want to see.

What this means is that while the assignment of a conceptual data modeler is to devise structures that will provide for all possible data structures, the decisions of the data mart designer reflect the specific, often arbitrary, wishes of particular users of the system. There is no "right" design for a data mart, other than that which your user demands. Among other things, this means that today's design, based on what the user *thinks* he wants today, may be wrong tomorrow. Among the technologies that are important here are those that allow you to change the architecture of the data mart easily.

For example, while a conceptual model typically makes it possible to define any time period, a data mart may be encoded to look only at calendar months. Similarly, the corporate model may be sophisticated in describing all possible organizational structures, while the data mart may be encoded to see only departments and companies.

It may be the case that a company does not have a tidy conceptual model as the basis for running its operational systems. In this case, it is important to develop one, if only to be sure the true natures of all the data are properly understood. While requirements for a data mart come from particular requirements, it will be impossible to manage a set of data marts if the underlying structure of the data is not clear. It is a good idea to implement a variation on the ODS (call it a "central data warehouse") based on the conceptual model, even if the operational systems are not currently using one. Then the task of creating each data mart can be defined in terms of clearly understood concepts. Also, the task of retrieving data from the operational systems can be done less frequently (putting less of a load on those systems) and it does not have to be done whenever a data mart definition changes.

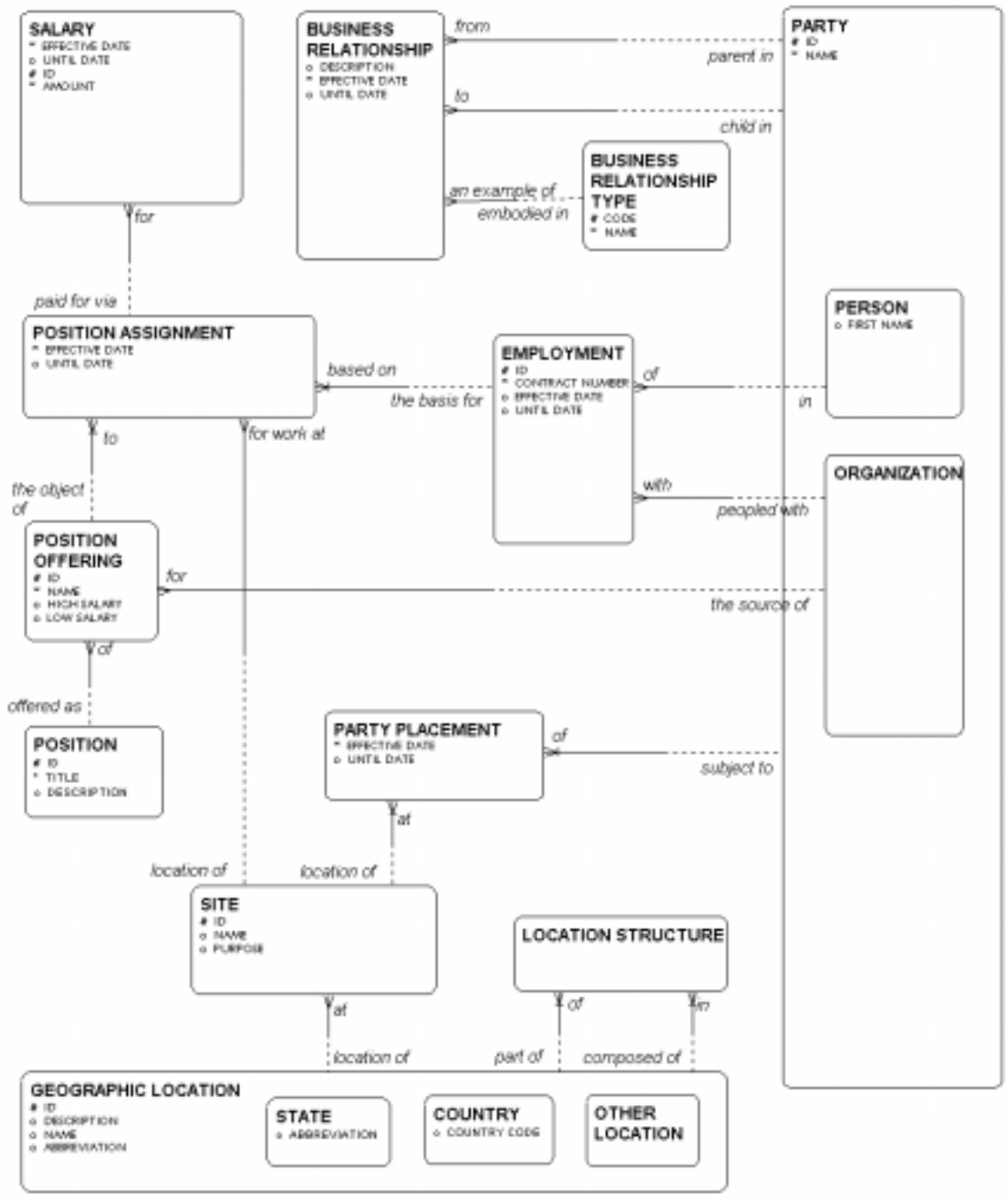



***Figure 2: Part of The Corporate Data Model***

## Dimensions and Facts

Data marts not only do not have to be normalized, they *should not* be if they are to satisfy the retrieval requirements. A table in the data mart may collapse several entities from the conceptual model.

The model of the data mart is usually in terms of the "dimensions" of the data – the terms of reference, by which retrievals are expected to be specified. The data could be considered to be in a cube, with, for example, with each data element at the intersection of the dimensions "product", "time", and "geography."

The data to be retrieved are collapsed into one or more "fact tables", related to each of the dimensions. An entity/relationship diagram can then be drawn of each fact and its dimension tables, with each occurrence of a fact entity related to one and only one of each of the dimensions. With the fact entity in the center and the dimension entities distributed around it, it tends to look like a star. For this reason, the design is often called a "star schema".

Since the data organization is in terms of these dimensions, the data base implementing this model is often called a "multi-dimensional data base".

With this organization, queries may consist simply of requesting all the facts associated with specified values of the dimensions. If a fact table describes sales, you can request sales for product x, geography y, and time z.

The dimensions themselves may be hierarchical. A product is part of a group, which is part of a product line; a month is part of a quarter, which is part of a year; a city is part of a state, which is part of a region; and so forth. In its simplest form, each occurrence of the lower level dimension is simply identified by attributes identifying its membership in the higher levels. To request data for a region means selecting the set of geography dimension rows that are in that region, and then finding the facts that are related to those rows.

Some will give the higher level dimensions tables in their own right, providing a multi-level star, or "snowflake schema".

## Multi-dimensional Technology

Oracle last year acquired Oracle Express, which is purported to be a data warehouse database manager. It is a tool whose underlying architecture reflects this multi-dimensional view of the world. Presumably multi-dimensional database technology includes a language specifically geared to describing queries in multi-dimensional terms.

Such a tool is not required, however, to take advantage of the multi-dimensional approach. It is perfectly reasonable to define tables to capture this information in a relational database. It is not clear whether changing the technology in addition to changing the data base design approach provides enough benefits to be worth the cost and aggravation.

## For Example . . .

The generality of the corporate model makes it much too abstract for the average data user. Business people want data in their terms and they are quite happy to constrain their model to a very specific set of things. We can imagine the following example, derived from the conceptual model in Figure 2:

- All time references are to a month, or a multiple of months.
- The geography of interest is primarily state, although we sometimes are also interested in country. We don't care about the particulars of offices or other facilities in the state. We are only concerned with states in the United States.
- We want to know how many employees there are, by position and by department and company, in various categories, but we don't care about them individually.
- The only data that concern us today are salaries.

With these specifications, we can construct a data mart. We start by identifying the source of the fact entity. Our fact of interest is "salary", which is an attribute of SALARY.

The dimensions of interest are:

- Month
- State and country
- Division and company
- Position type

The dimension entities in most cases already exist, but some are camouflaged (See Figure 3):

- POSITION can be used pretty much in its present form. The attributes of POSITION OFFERING and POSITION ASSIGNMENT will be subsumed into the fact entity, salary.
- A DEPARTMENT is a kind of ORGANIZATION. The appropriate ORGANIZATIONS will have to be selected to create a DEPARTMENT dimension entity. Note that there are two paths to SALARY from ORGANIZATION. It will be necessary to find out from your user whether the DEPARTMENT associated with a SALARY is the hiring department or the department where each person works. If it is the hiring department, the attributes of EMPLOYMENT and POSITION ASSIGNMENT must be subsumed into the SALARY fact entity.
- Note that if "Company" is to be an attribute of the DEPARTMENT dimension, it will be necessary to navigate through BUSINESS RELATIONSHIP to find out the company which owns each department.
- STATE is seen explicitly as a sub-type of GEOGRAPHIC LOCATION, but to determine the "Country" attribute for each STATE, it will be necessary to navigate through LOCATION STRUCTURE. Note that for SITES which are smaller than STATES, it is possible to add the facts together to arrive at numbers for them. Data regarding SITES that are only in COUNTRIES, however, will be lost to the data mart, because there is no way to attribute them to particular states.
- Note also that we have chosen not to deal with the location of the departments themselves. If a user wants to know in which states a department or company operates, that is not directly available — although it can be implied by determining where salaries are paid.
- The MONTH dimension will have to be created, with one occurrence for each calendar month. Each SALARY must then be linked to the appropriate MONTH occurrence, based on an algorithm applied to the "Effective date" and the "Until date".

Figure 4 shows the model of our data mart, based on these assertions. Note that the entity names have been prefaced by "DW" to distinguish them from conceptual model entities. DW SALARY contains the attributes not only of the original SALARY, but also of POSITION ASSIGNMENT, EMPLOYMENT, and PERSON. When implemented as a table, it will also acquire the foreign keys to DW MONTH, DW DEPARTMENT, DW STATE, and DW POSITION. DW DEPARTMENT has its attributes as an ORGANIZATION, plus those for its parent company. DW STATE, similarly, has acquired the attributes of its parent country.

Note also that one dimension of the original model — PERSON — has been eliminated. This means that DW SALARY occurrences are summarized, adding together the salaries of all people in each combination of position, department, state and month.

## Conclusions

Contrary to what many believe, the multi-dimensional model does not replace the conceptual, relational model. Indeed, the development of a normalized conceptual model will be critical to the success of any data warehouse project. It is impossible to manage a set of data marts without one.

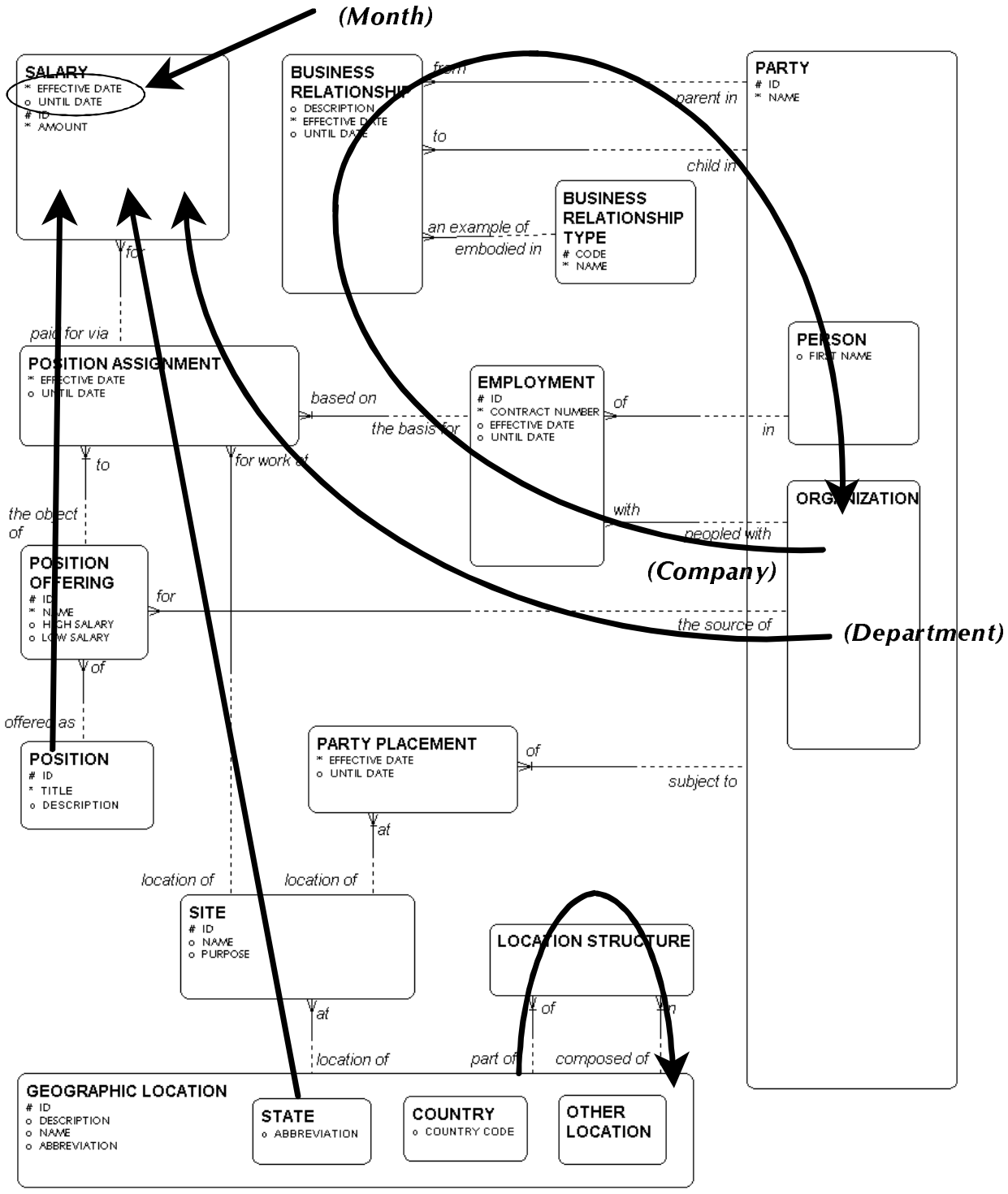
Instead, the development of multi-dimensional modeling techniques can be considered an extension to our data management tool kit. When you are developing the databases to be used for specific sets of retrievals, design them according to the user's perspective. That perspective will be constrained and concrete. It also will change over time.

The existence of a stable conceptual model will provide a basis for developing that constrained model — and will make it possible to react to those changes.

## About the author . . .

David C. Hay
Essential Strategies, Inc.

13 Hilshire Grove Lane
Houston, Texas 77055
(713) 464-8316
dhay@essentialstrategies.com








–

***Figure 3: Facts and Dimensions in the ODS***




–

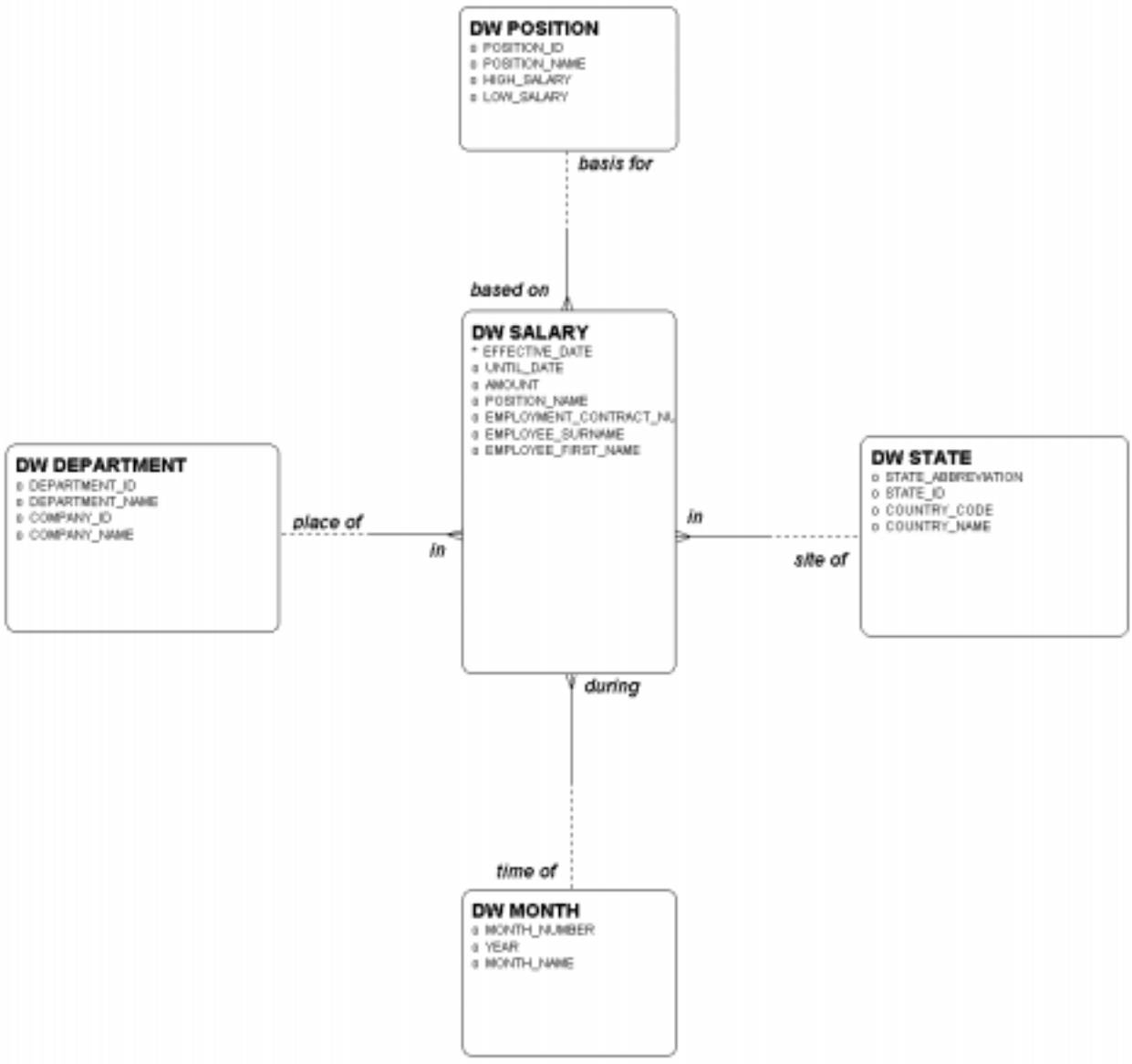



***Figure 4: The Multi-dimensional Data Model***




–