Table of Contents

[**What three (often conflicting) database requirements must be addressed in database design?** 1](#_Toc6040093)

[**Briefly, but precisely, explain the difference between single-valued attributes and simple attributes. Give an example of each.** 1](#_Toc6040094)

[**What are multivalued attributes, and how can they be handled within the database design?** 2](#_Toc6040095)

[**The Multivalued Attribute Solution** 3](#_Toc6040096)

[**The ERD for the CH3\_QUESTIONS Database** 4](#_Toc6040097)

# **What three (often conflicting) database requirements must be addressed in database design?**

Database design must reconcile the following requirements:

* Design elegance requires that the design must adhere to design rules concerning nulls, derived attributes, redundancies, relationship types, and so on.
* Information requirements are dictated by the end users.
* Operational (transaction) speed requirements are also dictated by the end users.

Clearly, an elegant database design that fails to address end user information requirements or one that forms the basis for an implementation whose use progresses at a snail's pace has little practical use.

# **Briefly, but precisely, explain the difference between single-valued attributes and simple attributes. Give an example of each.**

A single-valued attribute is one that can have only one value. For example, a person has only one first name and only one social security number. A simple attribute is one that cannot be decomposed into its component pieces. For example, a person's sex is classified as either M or F and there is no reasonable way to decompose M or F. Similarly, a person's first name cannot be decomposed into meaningful components. (In contrast, if a phone number includes the area code, it can be decomposed into the area code and the phone number itself. And a person's name may be decomposed into a first name, an initial, and a last name.)

Single-valued attributes are not necessarily simple. For example, an inventory code HWPRIJ23145 may refer to a classification scheme in which HW indicates HardWare, PR indicates Printer, IJ indicates InkJet, and 23145 indicates an inventory control number. Therefore, HWPRIJ23145 may be decomposed into its component parts... even though it is single-valued. To facilitate product tracking, manufacturing serial codes must be single-valued, but they may not be simple. For instance, the product serial number TNP5S2M231109154321 might be decomposed this way:

TN = state = Tennessee

P5 = plant number 5

S2 = shift 2

M23 = machine 23

11 = month, i.e., November

09 = day

154321 = time on a 24-hour clock, i.e., 15:43:21, or 3:43 p.m. plus 21 seconds

# **What are multivalued attributes, and how can they be handled within the database design?**

As the name implies, multi-valued attributes may have many values. For example, a person's education may include a high school diploma, a 2-year college associate degree, a four-year college degree, a Master's degree, a Doctoral degree, and various professional certifications such as a Certified Public Accounting certificate or a Certified Data Processing Certificate.

There are basically three ways to handle multi-valued attributes, and two of those three ways are

bad:

1. Each of the possible outcomes is kept as a separate attribute within the table. This solution is undesirable for several reasons. First, the table would generate many nulls for those who had minimal educational attainments. Using the preceding example, a person with only a high school diploma would generate nulls for the 2-year college associate degree, the four-year college degree, the Master's degree, the Doctoral degree, and for each of the professional certifications. In addition, how many professional certification attributes should be maintained? If you store two professional certification attributes, you will generate a null for someone with only one professional certification and you'd generate two nulls for all persons without professional certifications. And suppose you have a person with five professional certifications? Would you create additional attributes, thus creating many more nulls in the table, or would you simply ignore the additional professional certifications, thereby losing information?
2. The educational attainments may be kept as a single, variable-length string or character field. This solution is undesirable because it becomes difficult to query the table. For example, even a simple question such as "how many employees have four-year college degrees?" requires string partitioning that is time-consuming at best. Of course, if there is no need to ever group employees by education, the variable-length string might be acceptable from a design point of view. However, as database designers we know that, sooner or later, information requirements are likely to grow, so the string storage is probably a bad idea from that perspective, too.
3. Finally, the most flexible way to deal with multi-valued attributes is to create a composite entity that links employees to education. By using the composite entity, there will never be a situation in which additional attributes must be created within the EMPLOYEE table to accommodate people with multiple certifications. In short, we eliminate the generation of nulls. In addition, we gain information flexibility because we can also store the details (date earned, place earned, etc.) for each of the educational attainments. The (simplified) structures might look like those in Figure Q3.19 A-C.

# **The Multivalued Attribute Solution**

**Database name: CH3\_QUESTIONS**

**Table name: EMPLOYEE Table name: EMP\_EDUC Table name: EDUCATION**

FIG4-19AFIG4-19CFIG4-19B

FIG4-19D**The Relational Schema for the CH3\_QUESTIONS Database**

By looking at the structures shown in Figures Q3.19 A-C, we can tell that the employee named Randall earned a high school diploma in 1983, a Bachelor's degree in 1989, and a Certified Network Professional certification in 1992. If Randall were to earn a Master's degree and a Certified Data Processing certification later, we merely add another two records in the EMP\_EDUC table. If additional educational attainments beyond those listed in the EDUCATION table are earned by any employee, all we need to do is add the appropriate record(s) to the EDUCATION table, then enter the employee's attainments in the EMP\_EDUC table. There are no nulls, we have superb query capability, and we have flexibility. Not a bad set of design goals!

# **The ERD for the CH3\_QUESTIONS Database**

**The final four questions are based on the E-R diagram**