IDEA GROUP PUBLISHING
701 E. Chocolate Avenue, Hershey PA 17033-1117, USA
Tel: 717/533-8845; Fax 717/533-8661; URL-http://www.idea-group.com
ITB8000

## Chapter VIII

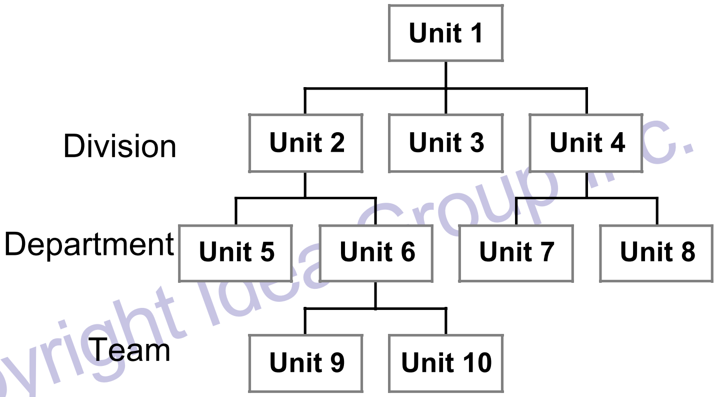
# Accommodating Hierarchies in Relational Databases

Ido Millet
Penn State Erie, USA

## INTRODUCTION

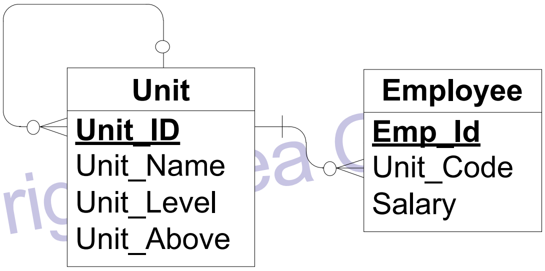
Relational databases and the current SQL standard are poorly suited to retrieval of hierarchical data. After demonstrating the problem, this chapter describes how two approaches to data denormalization can facilitate hierarchical data retrieval. Both approaches solve the problem of data retrieval, but as expected, come at the cost of difficult and potentially inconsistent data updates. This chapter then describes how we can address these update-related shortcomings via back-end (triggers) logic. Using a proper combination of denormalized data structure and back-end logic, we can have the best of both worlds: easy data retrieval and simple, consistent data updates.

*Figure 1. A typical unit hierarchy*



This chapter appears in the book, *Developing Quality Complex Database Systems: Practices, Techniques and Technologies* by Shirley Becker. Copyright © 2001, Idea Group Publishing.

*Figure 2. A normalized data model with a unit hierarchy*



Hierarchies occur in a variety of application areas such as biology, geography, manufacturing, accounting, knowledge management, marketing and human resources. Consider, for example, the organizational unit hierarchy depicted in Figure 1.

Since each unit has at most one parent, we can easily implement this hierarchy via a recursive relationship. This means that each unit record can maintain a foreign key pointing at the unit above it. Figure 2 shows a data model for this situation, including a relationship between Unit and Employee entities.

It should be noted that for many situations involving data hierarchies, flexibility can be gained by adding a lookup table that maintains the domain-specific name for each level. This allows the same data model to serve different needs. For example, some organizations may refer to units at level 2 as Divisions, while others may refer to such units as Departments, Groups, Branches, Stores or Offices. A hierarchy level lookup table allows easy adaptation of the same system to different contexts and different organizations.

To demonstrate the difficulty of hierarchical data retrieval against the normalized data model in Figure 2, consider the following requests:

- Show a list of all units under unit 2
- Show a list of all employees in or under a given division
- Show how many employees work in or under each division
- Show total employee salaries for each division including all units below it

Using SQL, we can easily join each employee with her unit, but we cannot easily identify the division (parent unit at level 2) for each unit. This is because for each unit we know only the immediate parent. This difficulty in locating parent nodes at any given level is at the heart of the problem.

## SQL-BASED SOLUTIONS

While hierarchies pose a significant challenge for SQL, complex SQL can solve surprisingly tough problems. For example, the SQL statement in Listing 1 will return all units under division 2. The result set, shown in Figure 3, can then be established as a view and joined with employees to answer more complex queries.

The SQL:1999 standard (ANSI/ISO/IEC 9075-2-1999) pushes the limits of hierarchy data retrieval even further by supporting recursive queries. For example,