Chapter 16

A Table with a View:
Introduction to Database Concepts
"Computers are useless. They only give answers."
- Pablo Picasso
## A Spreadsheet Table

### A Standard Table

<table>
<thead>
<tr>
<th>Name</th>
<th>Assignment 1</th>
<th>Assignment 2</th>
<th>Assignment 3</th>
<th>Assignment Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice Johnson</td>
<td>90</td>
<td>80</td>
<td>70</td>
<td>240</td>
</tr>
<tr>
<td>Bob Kimball</td>
<td>40</td>
<td>50</td>
<td>60</td>
<td>150</td>
</tr>
<tr>
<td>Carlie Larsen</td>
<td>70</td>
<td>80</td>
<td>90</td>
<td>240</td>
</tr>
<tr>
<td>David McDonald</td>
<td>90</td>
<td>80</td>
<td>100</td>
<td>270</td>
</tr>
</tbody>
</table>

### A table with a shifted cell

<table>
<thead>
<tr>
<th>Name</th>
<th>Assignment 1</th>
<th>Assignment 2</th>
<th>Assignment 3</th>
<th>Assignment Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice Johnson</td>
<td>90</td>
<td>80</td>
<td>70</td>
<td>240</td>
</tr>
<tr>
<td>Bob Kimball</td>
<td>40</td>
<td>50</td>
<td>60</td>
<td>150</td>
</tr>
<tr>
<td>Carlie Larsen</td>
<td>70</td>
<td>80</td>
<td>90</td>
<td>240</td>
</tr>
<tr>
<td>David McDonald</td>
<td>90</td>
<td>80</td>
<td>100</td>
<td>270</td>
</tr>
</tbody>
</table>
A Database Table (XML)

<graderow>
<name>Alice Johnson</name>
<assignment1>90</assignment1>
<assignment2>80</assignment2>
<assignment3>70</assignment3>
<total>240</total>
</graderow>
create table grades (  
    name varchar(60),  
    assignment_1 unsigned int,  
    assignment_2 unsigned int,  
    assignment_3 unsigned int,  
    assignment_total unsigned int  
);
Spreadsheets vs. Databases

Spreadsheets are a nice way to organize data in a pretty table. Databases are even better because you can organize the data and identify exactly what the data is.
Database's Advantage

- Identify the type of data - Each different type of value is given a unique tag that can be used to identify it.
- Define the affinity of the data - Tags or separate tables make it possible to logically group related data.
What makes XML easy and intuitive is that there are no standard tags to learn. *We think up the tags we need!* Computer scientists call this a self-describing language, because whatever we create becomes the language (tags) to structure the data. (p. 446)
Attributes in XML

XML can have attributes just like HTML. It is best to use attributes for metadata, not for the actual data.

<score type="integer" value="89" />

vs

<score type="integer">89</score>
Effective Design with XML

- Identification Rule - Label data with tags consistently.
- Affinity Rule - Group related data.
- Collection Rule - Group related instances.
<school id="1">
  <name>University of Oregon</name>
  <address>1585 East 13th Avenue</address>
</school>

<school id="2">
  <name>Lane Community College</name>
  <address>4000 East 30th Avenue</address>
</school>
Relational databases describe the relationships among the different kinds of data - the sort of ideas embodied in Affinity and Collection Rules - allowing the software to answer queries about them. (p. 453)
Entities

Entities are pieces of data. Entities are things.
Instances are Unordered

In a database, each instance or row, is not ordered. In cases where order does matter, a column in a row could have a value that could be used to order the instances. The physical order of the columns does not matter either since that is all purely for display purposes.
In a table, no two rows should be exactly the same. Each row should be unique, but two different rows can both have the same value on one or more of their columns.
Keys are the columns that are designated to be unique. Any set of attributes or columns that make a unique set are called candidate keys. The creator of the table chooses one candidate key to designate as the primary key.
Atomic Data

Data is atomic if it cannot be divided into smaller pieces without losing meaning.
A database scheme or database schema is a collection of definitions that describe how the database is supposed to be organized.

It is common to use a database schema to validate the integrity of the database structure.
Table Operations

- **Select** - Filter rows from a table.
- **Project** - Filter columns from a table.
- **Union** - Combine two tables.
- **Difference** - Remove rows from one table that also exist in another table.
- **Product** - The product of the rows in two tables.
- **Join** - The intersection of two tables on a given match.
Structured Query Language (SQL)

- SELECT - Performs Select, Project, Difference, Product, and Join Operations.
- INSERT - Adds new rows to the database.
- DELETE - Removes rows from the database.
- UPDATE - Modifies existing rows in the database.
Physically, data is only stored in one place. This data can be backed up, but the data still only has one source that is used. Each time a user wants to view the data, they write a query that generates logical data that is neither stored nor updated.
Queries

Queries are commands that users sent to the database to request specific pieces of data.

Queries can be simple printouts of a tables contents, or complex combinations of tables and only select pieces of data.
In relational databases, tables are related and are usually marked or labeled in a way that makes it easier to identify relationships. One way that tables indicate that they are related is by sharing a common column name and values that can be used to match up and join the tables.
Grades Table(s) Revisited

CREATE TABLE student (  
    student_id char(9),  
    name varchar(60),  
    primary key (student_id)  
);

CREATE TABLE assignments (  
    student_id char(9),  
    assignment_id unsigned int,  
    assignment_grade unsigned int,  
    primary key (student_id, assignment_id)  
);
CREATE VIEW grades AS
SELECT student_id,
    sum(assignment_grade) as total
FROM assignments
GROUP BY student_id;
Grade Query

```
SELECT a.name, b.total
FROM student a,
     grades b
WHERE a.student_id = b.student_id;
```
Database Security

URL: http://xkcd.com/327/