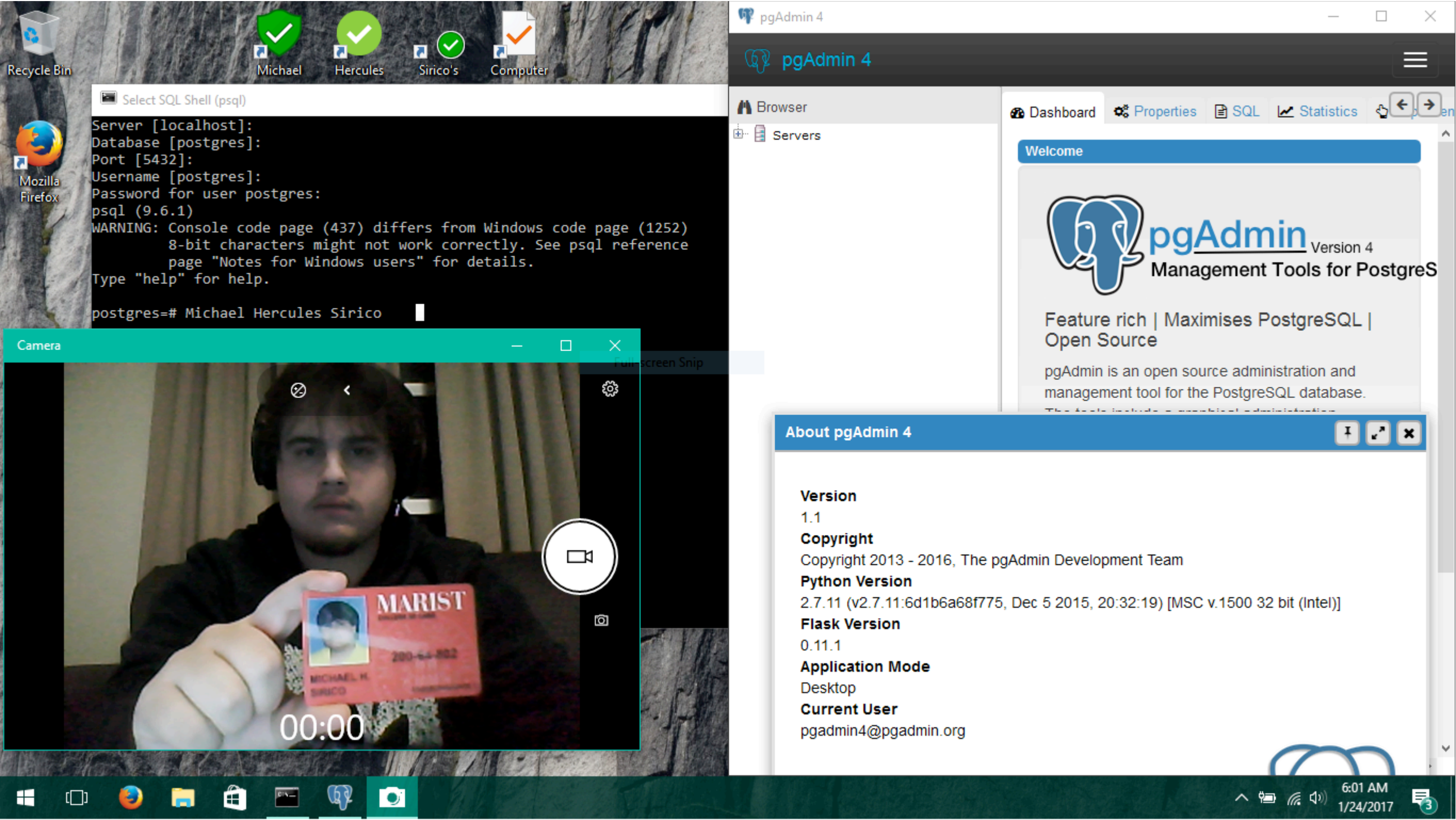
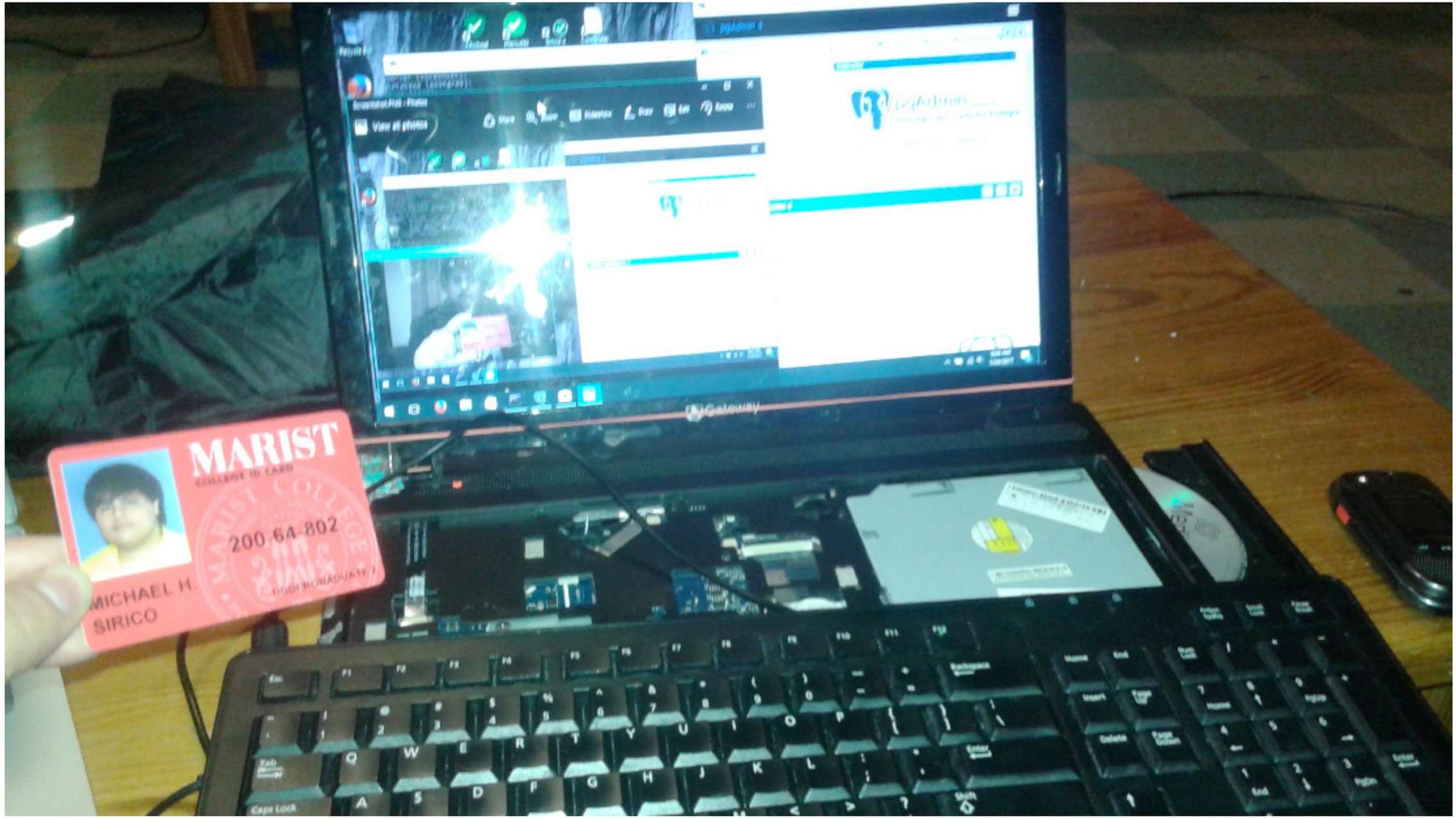

Recycle Bin
Michael
Hercules
Sirico's
Computer
Mozilla Firefox
Select SQL Shell (psql)
Server [localhost]:
Database [postgres]:
Port [5432]:
Username [postgres]:
Password for user postgres:
psql (9.6.1)
WARNING: Console code page (437) differs from Windows code page (1252)
8-bit characters might not work correctly. See psql reference
page "Notes for Windows users" for details.
Type "help" for help.
postgres=# Michael Hercules Sirico
Camera
MARIST
00:00
pgAdmin 4
Browser
Servers
Dashboard
Properties
SQL
Statistics
Welcome
pgAdmin
Version 4
Management Tools for PostgreS
Feature rich | Maximises PostgreSQL | Open Source
pgAdmin is an open source administration and management tool for the PostgreSQL database.
About pgAdmin 4
Version
1.1
Copyright
Copyright 2013 - 2016, The pgAdmin Development Team
Python Version
2.7.11 (v2.7.11:6d1b6a68f775, Dec 5 2015, 20:32:19) [MSC v.1500 32 bit (Intel)]
Flask Version
0.11.1
Application Mode
Desktop
Current User
pgadmin4@pgadmin.org
6:01 AM
1/24/2017


MARIST
COLLEGE ID CARD
200-64-802
MICHAEL H.
SIRICO
Gateway

Michael Hercules Sirico

Database Management

Alan Labouseur

January 24, 2017

Data vs. Information

After learning about MySQL Databases in Software Development II, I decided to put my skills to good use. I am secretary on the executive board of residence hall council for my area. Stored in the database are several columns and rows. Looking at this database without context would yield a lot of data; a large amount of numbers, and some string text. However, most of the numbers are meaningless without context; purely as data, these numbers are meaningless. An abbreviated example of a row in the database could be: [Michael Sirico, 20064802, 20, 8, 4, 3] Without context, it's difficult to determine what most of this means; those numbers are meaningless, as the observer does not know to what they refer without the appropriate context. However, adding context turns this meaningless data into information that can be analyzed and used. In this particular example of my Residence hall council database, The context comes in the form of Columns. These columns refer to the names of each member, the CWID number of each student, the id number in the database, the number of meetings they have attended, the number of events they have attended, and priority points earned. Upon viewing this data with context that has been given, the previously meaningless data becomes valuable information. We now know that "Michael Sirico" is the name of a member of Residence hall council, his CWID number is 20064802, His ID in the database is 20, he has attended 8 meetings, 4 events, and received 3 priority points. Knowing this context is essential to understanding the information in the database, because when lacking the context, the data is completely meaningless, and has little purpose or use. The context allows me, as secretary, to quickly look at any particular member of the residence hall council, and determine their standing in the group.

Data Models

A Hierarchical Data Model is a model where data has layers. As an example, there could be a model in which the top layer is Vehicles. Branching down from the Vehicles category, there is a Brands layer; with two brands. Each brand manufactures different types of vehicles, each brand branching to the layer below it, of vehicle type; Brand A has SUVs and Sedans, Brand B manufacturing Sedans and coupes. This presents several problems; There is Duplication; as both brand A and B have Sedans as a subcategory. A Network Model, which is not technically a hierarchy, is able to solve this problem, as rather than Brand A and B pointing to different copies of Sedans, they instead point to the same instance of Sedan. However; Yet another problem arises when the following is taken into consideration; “Station Wagon”, another Vehicle type exists in this data, and yet is not visible or shown, because neither brand A nor B happens to have that as a subgroup.

In Relational models, there is a much easier way to deal with problems like this. In Relational Models, there is an abstract of tables assembled by rows and columns. The First Table is a list of Car Brands; A and B. The Third Table is a list of all Vehicle Types; SUVs, Sedans, Coupes, and Station Wagons. The Second table, shows that Brand A in table 1 associates with SUVs and Sedans in table 3; and it also shows that Brand B in table 1 associates with Sedans and Coupes in table 3. In this way; There is no Duplication, and there is no Hidden Data. All Manufacturers are listed in table 1, and all vehicle types are listed in table 3; and table 2 shows that relationship. Relational Models allow for Cardinality; relation between two table entities can be one-to-one, 1-to-many, or many-to-many. In The situation of Many-to-many, using an intermediate table with the relations is important; table 2 in the example given. In addition, by definition, inconsistencies are not allowed, which makes the data and those relationships more straightforward.

Considering the significant Pros of the Relational model in storing data, and the very apparent flaws and shortcomings of the hierarchical model, The inherent Hierarchical Structure of XML would seem to be a less than efficient model of data storage.

Solution to Maze

1) Down [Red]

2) Right [Yellow]

3) Down [Blue]

4) Right [Red]

5) Down/Right [Yellow]

6) Up [Blue]

7) Up [Red]

8) Left [Yellow]

9) Down [Blue]

10) Left [Red]

11) Down Right [Yellow]

12) Right [Blue] **exit**