Introduction to Databases

INFO 344 Winter 2007

...or

The Day Your Instructor Finally Thought of a Way for the Students in the Back to See These Slides!

http://faculty.washington.edu/loter/info344/slides/lect6_files/fullscreen.html

Administrivia

- Feedback from Project designs on Tuesday
- Lab 5 (Apache, SSI, htaccess) due today
- Questions?

Today

- Databases: persistent storage for our applications
- Database terminology
- Database queries

The problem

- We know how to make relatively complex applications
- However, we still don't have a way of managing persistent data on the server
  - Our code only maintains state as long as a script is running
  - Adding or manipulating data on the site requires us to use FTP
- Need a way to interact programmatically with data stored on the server

Possibilities

- Use *flat files* to store data on the server
  - Maintain directories of plain text or XML files
  - Use PHP's file I/O functions to read and write files
- Use a *relational database* to store data
  - Database server stores structured data used by our application
  - PHP code connects to database to retrieve or modify data

Introduction to Databases

Databases

- A *database* is any system that stores and organizes structured data
- A *database management system* (DBMS) is the software system that manages access to a database
  - Provides a user interface, query language, etc.
- A *relational database* is a type of database that consists of a number of tables ("entities") with relationships between them

Advantages of database systems

- Provides a structured system for storing data
- Standard tools exist for interacting with relational databases (RDBs); no proprietary formats
- DBMSes provide sophisticated tools for manipulating data
- Database can effectively match our domain model
  - PHP Book class becomes Book table, etc.

The simplest database

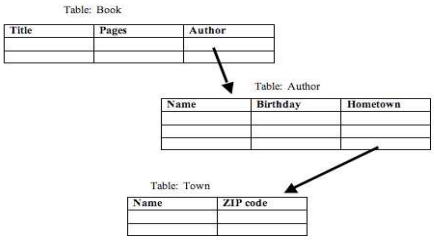
	A	B	C	D
1	**Author**	**Title**	**Pages**	**ISBN**
2				
3	Steven Holzner	Spring into PHP 5	600	1234567890
4	Joseph Fakename	All About Spinach	220	3728104554
5	Douglas Hofstadter	Gödel, Escher, Bach	1200	3141592653
6	Dr. Seuss	Horton Hears a Who	32	9746378282
7				

Terminology

- A *table* is a set of rows and columns that represent a particular type
  - e.g. Book, Student, NewsItem
- An *entity* is a specific row in the table that represents an instance of that type
  - e.g. "Horton Hears a Who", Betsy Smith
- Each entry has a number of attributes defined within the table
  - e.g. ISBN, student ID, birthdate
- A *database instance* is a self-contained set of tables, e.g. for an application

Relations

- Entities may be related to one another
  - Attributes in one table may reference another table



Database modeling

- A *database schema* is a description of the database design
  - Entities and their attributes
  - Relationships between entities
  - Types of relationships: one-to-one, one-to-many, etc
- Once again, we will strive for well-structured databases
  - Minimize redundancy and inconsistency
  - Simple and easy to understand
  - Effectively model real-world domain

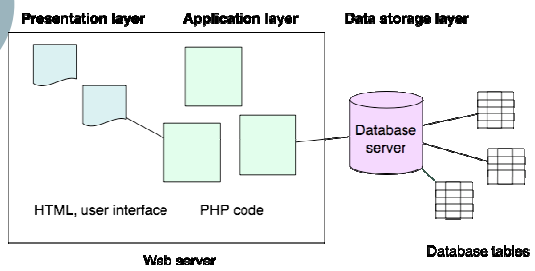
13

Defining entities

- What should our entities be?
- In our case, entities will often map to our model objects
  - NewsItem class => NewsItem table
- Generally, keep domain objects in their own tables and link them together
  - Rather than having one big table
- We may require additional tables to represent relations between entities

14

Database application model



15

Working with MySQL

16

Working with MySQL

- MySQL is a free, open source database management system
- We'll use a database server provided by the iSchool
  - But you can set up your own MySQL at home

17

Interacting with with databases

- Creating queries with SQL
- Database interfaces
- Queries for defining data
- Queries for manipulating data

18

Structured Query Language (SQL)

- Interaction with the server occurs in the form of queries
- A query describes an operation that we wish to perform on the database
  - Create, Read, Update, Delete (CRUD)
- Queries may return a table of values
  - But not always
- MySQL queries are written in SQL

19

Database interfaces

- MySQL provides a number of methods for handling queries
- mysql – command-line query tool
- phpMyAdmin – web based front end
- Code libraries in PHP and other languages

20

Creating databases and tables

- Before creating tables, we must create a database instance to hold them

```
CREATE DATABASE `library`;
```

- We use the CREATE TABLE statement to create a table
  - Specify the table's attributes, their types and constraints on allowable data values

21

CREATE TABLE example

```
CREATE TABLE `Books` (
    id int NOT NULL auto_increment,
    title varchar(255) NOT NULL,
    author varchar(255) NOT NULL,
    pages int NOT NULL,
    isbn varchar(10) NOT NULL,
    PRIMARY KEY  (id)
)
```

22

SQL types

- SQL supports a large number of data types
- We're only concerned with a few
  - INT – integer
    - Also BIGINT, MEDIUMINT, SMALLINT
    - TYPED and UNTYPED
  - DECIMAL – decimal number (like float)
  - VARCHAR(n) – string of up to size *n*
  - TIMESTAMP – date and time
  - TINYINT – small integer value, use 0 or 1 for boolean values

see http://dev.mysql.com/doc/refman/5.0/en/data-types.html

23

A Note About Data Types

- With scripting languages, we haven't had to worry about this
  - Javascript & PHP are weakly-typed
- Data types in databases are required
- Why so many?
  - Space in each record is allocated according to the data type
  - Cf. the Y2K Problem
  - Also, recent snafu on Slashdot
  - http://slashdot.org/articles/06/11/09/1534204.shtml

24

Primary keys

- Usually, we want to assign a unique identifier to each entity/row in our database
- An attribute that uniquely identifies a row in a database is known as a *primary key*
- We can tell MySQL to create and manage a primary key for us
  - Creates unique integer value for each row

25

Creating a primary key

```
CREATE TABLE Books (
    id int(11) NOT NULL
    auto_increment,
    title varchar(255) NOT NULL,
    author varchar(255) NOT NULL,
    pages int(11) NOT NULL,
    isbn varchar(10) NOT NULL,
    PRIMARY KEY  (id)
)
```

26

Foreign keys

- Often we want to have one table entity refer to another entity
- We can use the other entity's primary key as a reference in the table
  - This is called a *foreign key*

27

Foreign keys

id	Title	Author	Pages
7	The Sheltering Sky	3	340
8	Casino Royale	6	175

Table: Book

id	Name	Salary
3	Paul Bowles	500000
6	Ian Fleming	8000000

Table: Author

28

Separation in database design

- Generally it's a good idea to keep entity tables separate
  - Reduces the risk of redundant or inconsistent data
  - Use foreign keys to maintain relationships

id	Title	Author	Pages
7	The Sheltering Sky	**Paul Bowles**	340
8	Casino Royale	Ian Fleming	225
25	Dr. No	**Ian Flemming**	455

29

Data operations with SQL

- Any complete database will support the following operations, called CRUD
- Create records (SQL INSERT)
- Read records (SQL SELECT)
- Update records (SQL UPDATE)
- Delete records (SQL DELETE)

30

INSERT

- Adds a row to the table
- Syntax:
  INSERT INTO *table* (*param1, param2*)
  VALUES(*val1, val2*);
- Example:
  INSERT INTO Books (title,author,pages)
  VALUES ("info 344 notes","jim","15");

31

SELECT

- Retrieves rows and columns from the table
- Syntax:
  SELECT *attributes* FROM *table* WHERE *conditions*;
- Example:
  SELECT title, pages FROM Books WHERE pages = 200;
- Use SELECT * to return all columns

32

SELECT Example

SELECT title, pages FROM Books
WHERE pages = 200;

title	pages
Moby Dick	200
1984	200

33

WHERE clause

- Provide conditions to filter the results
- Can join clauses with AND and OR
- Comparators: =, >, >=, <, <=, <> (not equal)

  SELECT * FROM Books WHERE pages > 200 AND pages < 400;

34

Selecting attributes from multiple tables

- We can use SELECT and WHERE to join tables together with matching keys
  - Allows us to join related tables together

SELECT Books.titles, Books.author, Author.id, Author.name FROM Books, Author WHERE Book.author = Author.id;

title	author	id	name
The Sheltering Sky	7	7	Paul Bowles
Casino Royale	8	8	Ian Fleming

35

UPDATE

- Updates existing rows in the table
- Syntax:
  UPDATE *table* SET *param1 = value1, param2 = value2* WHERE *condition*;
- Example:
  /* fix misspelled names */
  UPDATE Author SET name = "Iam Fleming" WHERE name = "Ian Flemming";

36

DELETE

- Deletes rows from the table matching the specified criteria
- Syntax:
  DELETE FROM *table* WHERE *condition*;
- Example:
  /* remove Moby Dick from the catalog */
  DELETE FROM Books WHERE title = "Moby Dick";

37

PHP and MySQL

38

Connecting to MySQL from PHP

- PHP provides a number of functions for interacting with MySQL databases
  - mysql_connect()
  - mysql_db_query()
  - mysql_fetch_array()

39

mysql_connect()

```
$connection = mysql_connect($hostname, $username, $password);
```

- Creates a connection to the DBMS with the specified parameters
- Returns a DB connection object

```
$connection = mysql_connect("localhost", "jim", "mypassword");
```

40

mysql_connect() with error handling

- Additional handling for connection errors

```
mysql_connect("localhost", shaun, "mypassword") or die("Could not connect: " . mysql_error());
```

41

mysql_db_query()

```
$result = mysql_db_query($databasename, $query, $connection);
```

- Uses the specified connection and database name to execute the query
- Stores results inside result object

```
/* gets all books in the library database and Books table */
$result = mysql_db_query("library", "select * from Books", $connection);
```

42

mysql_fetch_array()

```
$row = mysql_fetch_array($result);
```

- Returns the next row from the query results as an associative array
- Returns FALSE if there are no more rows

```
/* gets all books in the library database
and Books table */
$result = mysql_db_query("library"
```


Example

```
<?php
   // connect to the DBMS
   $connection = mysql_connect("localhost", "jim",
   "mypassword");
   // select all books from the Library database
   $result = mysql_db_query("library", "select * from
   Books", $connection);
   // loop through each book
   while($row = mysql_fetch_array($result); {
        echo $row["title"]; // echo the title for each book
   }
?>
```