PHP 04

Databases

MySQL

PHPMyAdmin

PHP and MySQL

Peter Cho | 161B | Winter 06

Based on a tutorial by Prof. Daniel Sauter; Rasmus Lerdorf & Kevin Tatroe: Programming PHP. Sebastopol: O'Reilly, 2002; David Lane, Hugh E. Williams: Web Database Application with PHP and MySQL, 2nd Edition. Sebastopol: O'Reilly, 2004; and http://php.net

Databases are useful!

Most of the services we use on the Web are provided by web database applications. Web-based email, online shopping, forums and bulletin boards, corporate web sites, and news portals are all database-driven.

The use of databases has several potential advantages.

– separation of design and content, by working with templates

– content often outlasts the design of a Web site

– search and sort capabilities (access to all columns of a DB)

– easy backup and recovery

Databases

PHP supports over 20 types of databases, both commercial and open source.

In this class we are focusing on the MySQL relational database system, using the Structured Query Language (SQL) to communicate with the database.

In a Database Management System (DBMS), running on a database server, the data is structured into tables where each table has some number of columns, each of which has a name and a type (e.g. one table might keep track of all purchased items in an e-business where another table stores the billing and shipping address of the customer, connected through a key)

Database terms

Database

A repository to store data. For example, a database might store all of the data associated with finance in a large company, information about your CD and DVD collection, or the records of an online store.

Table

A part of a database that stores data related to an object, thing, or activity. For example, a table might store data about customers. A table has columns, fields, or attributes. The data is stored as rows or records.

Attributes

The columns in a table. All rows in a table have the same attributes. For example, a *customer* table might have the attributes *name*, *address*, and *city*. Each attribute has a data type such as string, integer, or date.

Rows

The data entries stored in a table. Rows contain values for each attribute. For example, a row in a *customer* table might contain the values “Matthew Richardson,” “Punt Road,” and “Richmond.” Rows are also known as records.

Relational model

A formal model that uses database, tables, and attributes to store data and manages the relationship between tables.

Database terms, cont.

(Relational) database management system (DBMS)

A software application that manages data in a database and is based on the relational model. Also known as a database server.

SQL

A standard query language that interacts with a database server. SQL is a set of statements to manage databases, tables, and data. Despite popular belief, SQL does not stand for Structured Query Language and isn't pronounced Sequel: it's pronounced as the three-letter acronym S-Q-L and it doesn't stand for anything.

Primary key

One or more attributes that contain values that uniquely identify each row. For example, a *customer* table might have the primary key named `cust_ID`. The `cust_ID` attribute is then assigned a unique value for each customer. A primary key is a constraint of most tables.

Index

A data structure used for fast access to rows in a table. An index is usually built for the primary key of each table and can then be used to quickly find a particular row. Indexes are also defined and built for other attributes when those attributes are frequently used in queries.

Database table

Winery Table

'attributes' (also 'fields' or 'columns')

Winery ID	Winery name	Address	Region ID
1	Moss Brothers	Smith Rd.	3
2	Hardy Brothers	Jones St.	1
3	Penfolds	Arthurton Rd.	1
4	Lindemans	Smith Ave.	2
5	Orlando	Jones St.	1

'row'
(also, 'record')

Relational database

Table

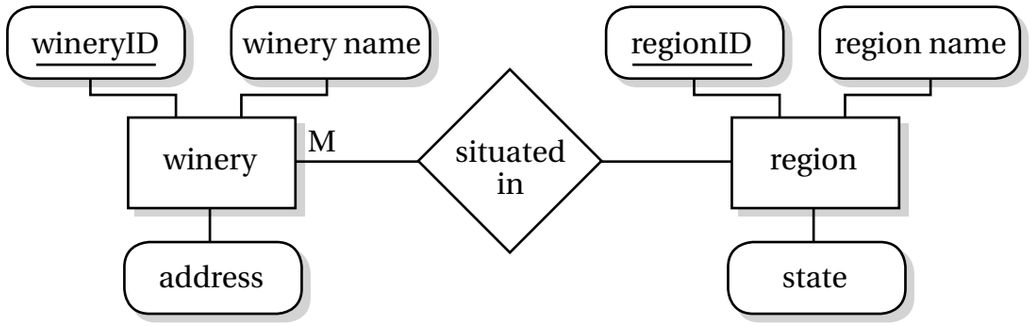
Winery Table

Winery ID	Winery name	Address	Region ID
1	Moss Brothers	Smith Rd.	3
2	Hardy Brothers	Jones St.	1
3	Penfolds	Arthurton Rd.	1
4	Lindemans	Smith Ave.	2
5	Orlando	Jones St.	1

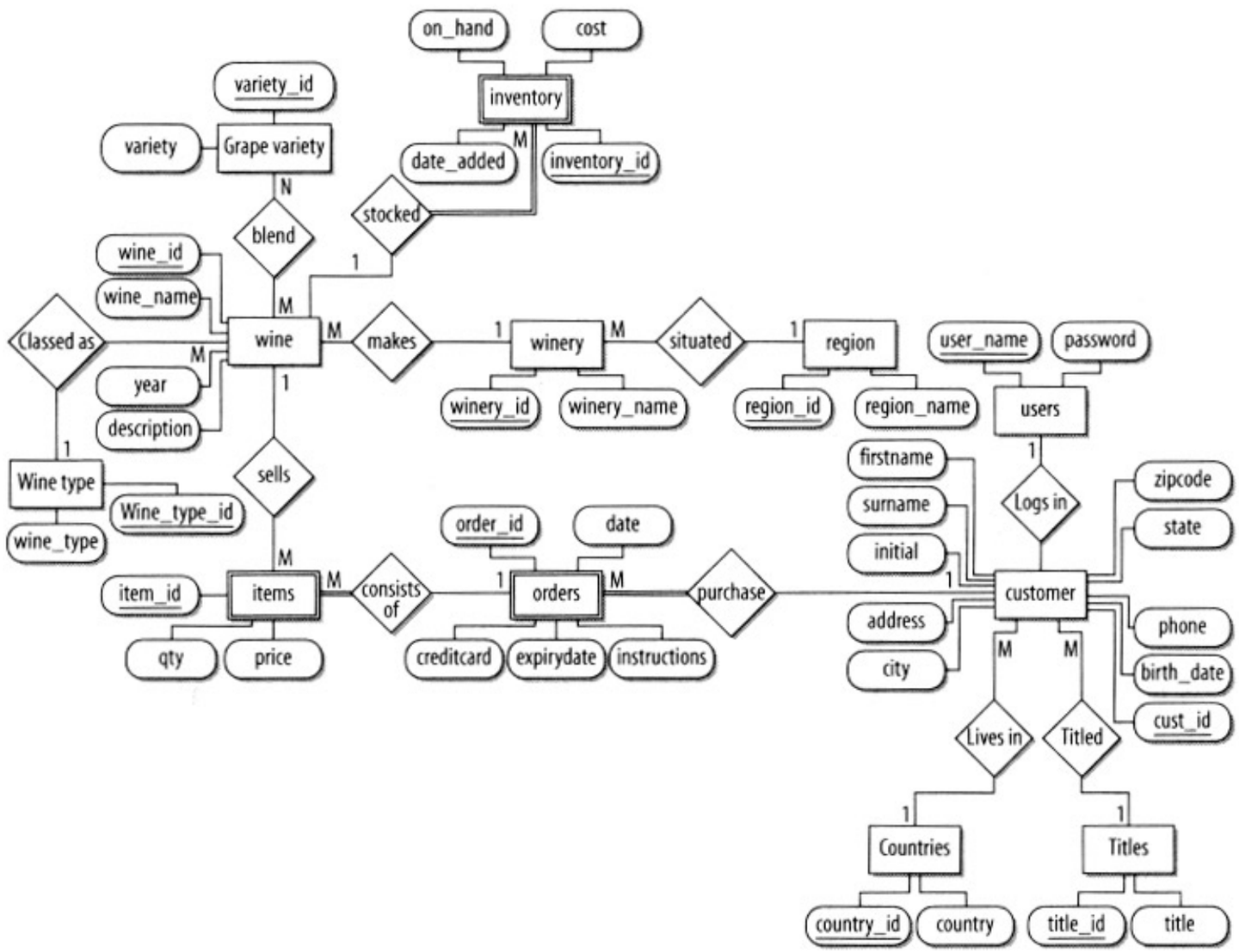
Region Table

Region ID	Region name	State
1	Barossa Valley	South Australia
2	Yarra Valley	Victoria
3	Margaret River	Western Australia

ER (entity-relationship) model



A more complex entity-relationship model



users.design.ucla.edu >> localhost >> petercho >> City | phpMyAdmin 2.6.0-rc1 - Mozilla Firefox

File Edit View Go Bookmarks Tools Help

https://users.design.ucla.edu/phpMyAdmin/

phpMyAdmin

petercho (7)
- book
- City
- Country
- CountryLanguage
- section_management
- words
- work

Server: localhost ▸ Database: petercho ▸ Table: City

Structure | Browse | SQL | Search | Insert | Export | Operations | Empty | Drop

	Field	Type	Collation	Attributes	Null	Default	Extra	Action
☐	ID	int(11)			No		auto_increment	
☐	Name	char(35)	latin1_swedish_ci		No			
☐	CountryCode	char(3)	latin1_swedish_ci		No			
☐	District	char(20)	latin1_swedish_ci		No			
☐	Population	int(11)			No	0		

Check All / Uncheck All *With selected:*

Print view | Propose table structure

Add new field: 1 At End of Table Go

Indexes:

Keyname	Type	Cardinality	Action	Field
PRIMARY	PRIMARY	4079		ID

Create an index on 1 columns Go

Space usage:

Type	Usage	
Data	273,293	Bytes
Index	35,840	Bytes
Total	309,133	Bytes

Row Statistic:

Statements	Value
Format	fixed
Collation	latin1_swedish_ci
Rows	4,079
Row length ø	67
Row size ø	76 Bytes
Next Autoindex	4,080
Creation	Jan 27, 2006 at 05:45 PM
Last update	Jan 27, 2006 at 05:46 PM

Run SQL query/queries on database petercho

```
SELECT * FROM `City` WHERE 1
```

Fields:
ID
Name
CountryCode
District
Population

<< Go

☑ Show this query here again

***Or* Location of the textfile:**

Location of the textfile:

Browse... (Max: 8,192KB)

Compression:
◉ Autodetect ○ None ○ "gzipped" ○ "bzipped"

Go

Insert data from a textfile into table

Done users.design.ucla.edu

web-based admin for your database

MySQL 4.1.12

Creating tables with SQL

```
CREATE TABLE customer (
    cust_id int(5) NOT NULL,
    surname varchar(50),
    firstname varchar(50),
    initial char(1),
    title_id int(3),
    address varchar(50),
    city varchar(50),
    state varchar(20),
    zipcode varchar(10),
    country_id int(4),
    phone varchar(15),
    birth_date char(10),
    PRIMARY KEY (cust_id)
) type=MyISAM;
```

Common SQL data types

more at http://dev.mysql.com/doc/, Ch. 11

`int(`*length*`)`	Integer with a max length; for IDs, age, counters, etc.
`decimal(`*width* `[,`*decimal_digits* `])`	a number with a *width* including an optional number of *decimal_digits* after the decimal point; used for currency, measurements, etc.
`datetime`	stores a date and time in the format YYYY-MM-DD HH:MM:SS
`time`	stores a time in the format HH:MM:SS
`date`	stores a date in the format YYYY-MM-DD
`timestamp`	stores the date and time in the format YYYYMMDDHHMMSS
`varchar(`*length*`)`	unpadded, variable-length text string with a specific maximum *length*
`char(`*length*`)`	padded, fixed-length text string of size *length*
`blob`	stores up to 64 KB of data

Basic SQL statements

```
// Creating a new entry (row) in a table
INSERT INTO items VALUES (0, 'screwdriver', 293848, 29.95, '04-12-01')

// Deleting a row in a table
DELETE FROM items WHERE number=223344

// Updating values in a specific row or multiple rows
UPDATE items SET date='05-01-12' where id=0

// Reading out rows where the condition is true
SELECT * FROM items WHERE date >= '04-08-01' AND price <= 50

// Reading out specific fields/values where the condition is true
SELECT items.title, items.price, customers.firstName,
customers.lastName, customer.zipCode WHERE items.number=293848
```

Connecting to the MySQL server with PHP

```
// server connect
$host = 'users.design.ucla.edu';
$usr = 'petercho';
$pwd = 'myPassword';
$db = 'petercho';

mysql_connect($host, $usr, $pwd) or die(mysql_error());
mysql_select_db($db);
```

SELECT statement

SELECT is used to retrieve rows selected from one or more tables.

```
$news = mysql_query("SELECT id, date, title, text, url
FROM upcoming ORDER BY date");

$news = mysql_query("SELECT id, date, title, text, url
FROM upcoming WHERE title="Talk" ORDER BY date DESC");
```

SELECT statements

```
SELECT surname, firstname FROM customer;
+-----------+-----------+
| surname   | firstname |
+-----------+-----------+
| Marzalla  | Dimitria  |
| LaTrobe   | Anthony   |
| Fong      | Nicholas  |
| Stribling | James     |
+-----------+-----------+
4 rows in set (0.04 sec)

SELECT * FROM region;
+-----------+---------------------+
| region_id | region_name         |
+-----------+---------------------+
|         1 | All                 |
|         2 | Goulburn Valley     |
|         3 | Rutherglen          |
|         4 | Coonawarra          |
|         5 | Upper Hunter Valley |
|         6 | Lower Hunter Valley |
|         7 | Barossa Valley      |
|         8 | Riverland           |
|         9 | Margaret River      |
|        10 | Swan Valley         |
+-----------+---------------------+
10 rows in set (0.01 sec)
```

SELECT statements, cont.

```
SELECT * FROM region WHERE region_id <= 3;
+-----------+-----------------+
| region_id | region_name     |
+-----------+-----------------+
|         1 | All             |
|         2 | Goulburn Valley |
|         3 | Rutherglen      |
+-----------+-----------------+
3 rows in set (0.03 sec)
```

```
SELECT region_name FROM region WHERE region_id <= 3;
+-----------------+
| region_name     |
+-----------------+
| All             |
| Goulburn Valley |
| Rutherglen      |
+-----------------+
3 rows in set (0.01 sec)
```

```
SELECT * FROM customer WHERE surname='Marzalla' AND
firstname='Dimitria';
```

```
SELECT cust_id FROM customer WHERE (surname='Marzalla' AND firstname
LIKE 'M%') OR birth_date='1980-07-14';
```

LIMIT

The LIMIT operator is used to control the size of the output.
Row numbering begins at row zero.

```
// returns only the first five rows from the customer table

SELECT * FROM customer LIMIT 5;

// returns the 100th to 104th rows from the customer table

SELECT * FROM customer LIMIT 100,5;

// set the second parameter to -1 to get all rows after a particular row

SELECT * FROM customer LIMIT 600,-1;
```

Reading out values (PHP)

```
while (list($id, $date, $name, $statement, $url) = mysql_fetch_row($webstudent)) {
        echo "ID: $id <br>";
        echo "DATE: $date <br>";
        echo "NAME: $name <br>";
        echo "STATEMENT: $statement <br>";
        echo "URL: $url <br>";
}
```

Inserting/adding rows (PHP)

```
$insert_webstudent = "INSERT webstudent (date, name, statement, url)
   VALUES ('$date', '$name', '$statement', '$url')";
mysql_query ($insert_webstudent);
```

Deleting rows (PHP)

```
$delete_webstudent = "DELETE FROM webstudent WHERE id = '$update_id'";
mysql_query ($delete_webstudent);
```

Updating rows (PHP)

```
$update_webstudent = "UPDATE webstudent SET date = '$date',
   name = '$name', statement = '$statement', url = '$url'
   WHERE id = '$update_id'";
mysql_query ($update_webstudent);
```

MySQL resources

Please refer to http://dev.mysql.com/doc/
for a detailed MySQL Reference.

Also see http://us3.php.net/manual/en/ref.mysql.php
for MySQL functions in PHP.

O'Reilly offers a variety of books on this subject available
online through the UCLA proxy server.