

MySQL

Dr. Michele Weigle

<http://www.cs.odu.edu/~mweigle/CS418-F12/>

Outline

- ▶ Assigned Reading
 - ▶ Chapter 3 “Using PHP5 with MySQL”
 - ▶ Chapter 10 “Building Databases”
 - ▶ Resource:
<http://dev.mysql.com/doc/>
- ▶ Quick review of relational databases
 - ▶ normalization
 - ▶ referential integrity
- ▶ Basic MySQL commands
- ▶ Ch 3 Code Example Demo/Walkthrough

Relational Database

- ▶ Collection of data organized in tables that can be used to create, retrieve, delete, and update that data in many ways

Keys

- ▶ Column where each item of data appears only once in that column
- ▶ Uniquely identifies the row
- ▶ *Primary key* – unique identifier for the table
- ▶ *Foreign key* – matches the primary key of another table

Super Hero Example

name	real name	power 1	power 2	power 3	lair address	city	state	zip
Clean Freak	John Smith	Strength	X-ray vision	flight	123 Poplar Ave	Townsburg	OH	45293
Soap Stud	Efram Jones	Speed			123 Poplar Ave	Townsburg	OH	45293
The Dustmite	Dustin Huff	Strength	Dirtiness	Laser Vision	452 Elm St. #3D	Burgtown	OH	45201

What if we need to add a super hero with more than 3 powers?

► 5

CS 418/518 - Fall 2012

1st Normal Form

name	real name	power 1	power 2	power 3	lair address	city	state	zip
Clean Freak	John Smith	Strength	X-ray vision	flight	123 Poplar Ave	Townsburg	OH	45293
Soap Stud	Efram Jones	Speed			123 Poplar Ave	Townsburg	OH	45293
The Dustmite	Dustin Huff	Strength	Dirtiness	Laser Vision	452 Elm St. #3D	Burgtown	OH	45201

- Eliminate repeating columns
- Add primary key to table
 - Unique
 - Must not change
- Maintain “atomicity“
 - Each cell is atomic, has only one item of data

► 6

CS 418/518 - Fall 2012

1NF

Result

- ▶ Eliminate repeating columns
- ▶ Add primary key to tables
- ▶ Each attribute is atomic

id	name	real name	power	lair address	city	state	zip
1	Clean Freak	John Smith	Strength	123 Poplar Ave	Townsborg	OH	45293
1	Clean Freak	John Smith	X-ray vision	123 Poplar Ave	Townsborg	OH	45293
1	Clean Freak	John Smith	flight	123 Poplar Ave	Townsborg	OH	45293
2	Soap Stud	Efram Jones	Speed	123 Poplar Ave	Townsborg	OH	45293
3	The Dustmite	Dustin Huff	Strength	452 Elm St. #3D	Burgtown	OH	45201
3	The Dustmite	Dustin Huff	Dirtiness	452 Elm St. #3D	Burgtown	OH	45201
3	The Dustmite	Dustin Huff	Laser Vision	452 Elm St. #3D	Burgtown	OH	45201

What if John Smith changes his name?

▶ 7

CS 418/518 - Fall 2012

2nd Normal Form

- ▶ Honor 1st Normal Form
- ▶ Create separate tables for data duplicated across rows
- ▶ Be aware of relationships!
 - ▶ 1:1
 - ▶ 1:m
 - ▶ m:n

id	name	real name	power	lair address	city	state	zip
1	Clean Freak	John Smith	Strength	123 Poplar Ave	Townsborg	OH	45293
1	Clean Freak	John Smith	X-ray vision	123 Poplar Ave	Townsborg	OH	45293
1	Clean Freak	John Smith	flight	123 Poplar Ave	Townsborg	OH	45293
2	Soap Stud	Efram Jones	Speed	123 Poplar Ave	Townsborg	OH	45293
3	The Dustmite	Dustin Huff	Strength	452 Elm St. #3D	Burgtown	OH	45201
3	The Dustmite	Dustin Huff	Dirtiness	452 Elm St. #3D	Burgtown	OH	45201
3	The Dustmite	Dustin Huff	Laser Vision	452 Elm St. #3D	Burgtown	OH	45201

▶ 8

CS 418/518 - Fall 2012

2NF

Result

id	lair_id	name	real name	align
1	1	Clean Freak	John Smith	good
2	1	Soap Stud	Efram Jones	good
3	2	The Dustmite	Dustin Huff	evil

id	lair address	city	state	zip
1	123 Poplar Ave	Townsburg	OH	45293
2	452 Elm St. #3D	Burgtown	OH	45201

id	power
1	Strength
2	X-Ray vision
3	Flight
4	Speed
5	Dirtiness
6	Laser Vision

char_id	power_id
1	1
1	2
1	3
2	4
3	1
3	5
3	6

- Satisfy 1NF
- Create separate tables for data duplicated across rows

Are “city” and “state” directly related to the lairs?

3rd Normal Form

- Honor 1st and 2nd Normal Form
- Create separate tables for any transitive or partial dependencies

id	lair address	city	state	zip
1	123 Poplar Ave	Townsburg	OH	45293
2	452 Elm St. #3D	Burgtown	OH	45201

3NF Result

id	lair_id	name	real name	align
1	1	Clean Freak	John Smith	good
2	1	Soap Stud	Efram Jones	good
3	2	The Dustmite	Dustin Huff	evil

id	zip_id	lair address
1	45293	123 Poplar Ave
2	45201	452 Elm St. #3D

id	city	state
45293	Townsburg	OH
45201	Burgtown	OH

id	power
1	Strength
2	X-Ray vision
3	Flight
4	Speed
5	Dirtiness
6	Laser Vision

char_id	power_id
1	1
1	2
1	3
2	4
3	1
3	5
3	6

- Satisfy 2NF
- Create separate tables for any transitive or partial dependencies

see note on p. 283 on why good/evil is not in a separate table

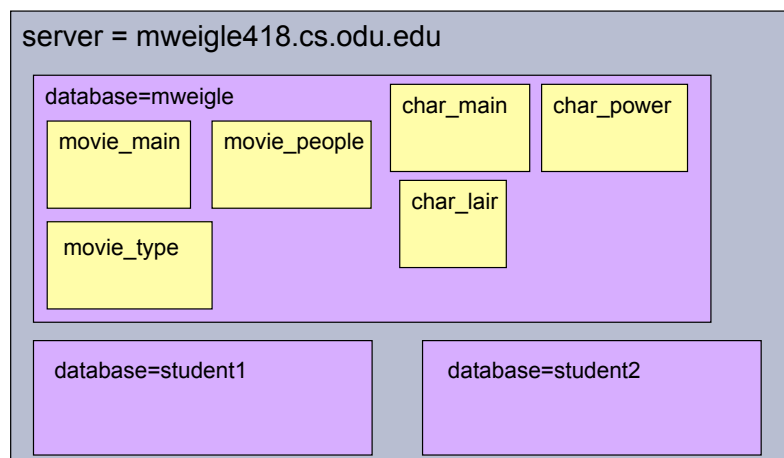
That's About as Far As We'll Go

- Other normal forms are possible (BCNF, 4NF, 5NF)
 - take a database class if you're interested
- Referential integrity
 - a foreign key ("link") into another table is no longer valid
 - "404 Errors" are *bad* in databases and should not happen
 - how bad is a function of the data itself...

Standardization

- ▶ **Table names**
 - ▶ descriptive – describe their main function and the application they belong to
 - ▶ relatively short
 - ▶ lowercase
- ▶ **Column names**
 - ▶ lowercase
 - ▶ short
 - ▶ separate words by ‘_’
- ▶ **Primary keys**
 - ▶ single primary keys always called ‘id’
- ▶ **Foreign keys**
 - ▶ end with ‘id’
 - ▶ start with table descriptor

MySQL Hierarchy For CS 418/518



We're sharing a single server, so each student has a single database (same as your username). Table names should start with the project or application name.

Manipulating Tables and Databases

- ▶ CREATE - create new databases, tables
- ▶ ALTER - modify existing tables
- ▶ DELETE - erase data from tables
- ▶ DESCRIBE - show structure of tables
- ▶ INSERT INTO *tablename* VALUES - put data in table
- ▶ UPDATE - modify data in tables
- ▶ DROP - destroys table or database (values + structure)

more: <http://dev.mysql.com/doc/refman/5.0/en/sql-syntax.html>

Native MySQL Data Types

- ▶ Unlike Perl, PHP and other civilized languages, MySQL is big into data types:
 - ▶ <http://dev.mysql.com/doc/refman/5.0/en/data-types.html>
 - ▶ pgs. 86-90 in Ch 3
 - ▶ many examples in Chs 3, 10

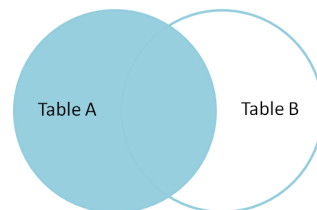
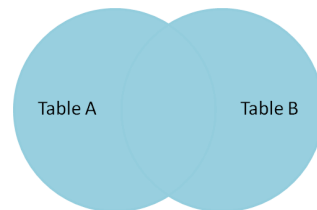
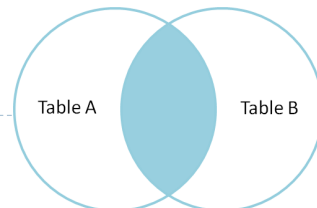
SQL Query Form

```
SELECT [fieldnames]
      FROM [tablenames]
      WHERE [criteria]
      ORDER BY [fieldname to sort on] [DESC]
      LIMIT [offset, maxrows]
```

more: <http://dev.mysql.com/doc/refman/5.0/en/select.html>
look at chapters 3 and 10 for code examples

SQL Joins

- Pull in data from two different tables
- INNER JOIN
 - find intersection between two tables
- FULL OUTER JOIN
 - produce set of all records in both tables
- LEFT JOIN (or, LEFT OUTER JOIN)
 - for each item in Table A, find some data in Table B



<http://www.codinghorror.com/blog/2007/10/a-visual-explanation-of-sql-joins.html>

PHP and MySQL

- ▶ `mysql_connect (“hostname”, “user”, “pass”)`
 - ▶ connect to the MySQL server
- ▶ `mysql_select_db (“database name”)`
 - ▶ makes the named database the active one
- ▶ `mysql_query (“query”)`
 - ▶ send any type of MySQL command to server
- ▶ `mysql_fetch_array (“results from query”)`
 - ▶ returns several rows of the query
- ▶ `mysql_error()`
 - ▶ shows the error message generated by the MySQL server

PHP and MySQL

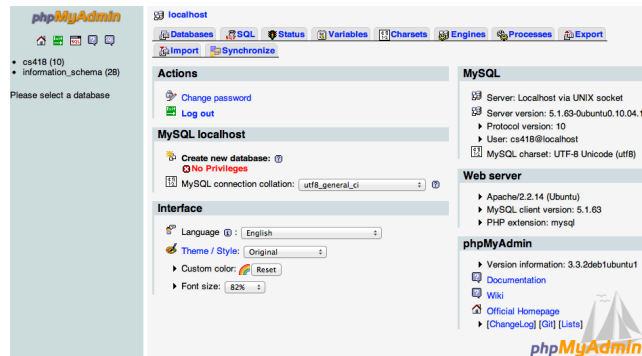
`config.php`

```
<?php
    define ('SQL_HOST', 'localhost');
    define ('SQL_USER', 'username');
    define ('SQL_PASS', 'passwd');
    define ('SQL_DB', 'username');
?>
```

- ▶ This way you don't have to include your plaintext username and password in your source files:
`require ('config.php');`
`mysql_connect (SQL_HOST, SQL_USER, SQL_PASS);`
`mysql_select_db(SQL_DB);`

phpMyAdmin

- ▶ Access your MySQL databases through a GUI
- ▶ Change your password
- ▶ Create, edit, delete tables
- ▶ Run (and test) queries
- ▶ View and print table structure



<https://mweigle418.cs.odu.edu/phpmyadmin>

Outline

- ▶ Quick review of relational databases
 - ▶ normalization
 - ▶ referential integrity
- ▶ Basic MySQL commands
- ▶ Ch 3 Code Example Demo/Walkthrough

Reminder:
Groups due Sep 6

Up Next: Tables, Forms
Assigned Reading: Chs 4, 5

Demo/Walkthrough Time

► Examples from Chapter 3

<http://www.cs.odu.edu/~mweigle/cs418/textbook/ch03.htm>

- Creating a Database
- Querying the Database
- PHP and Arrays of Data
- Multiple Tables

Movie Tables

movie_main

Field	Type
id	int(11)
name	varchar(255)
type	tinyint(2)
year	int(4)
lead_actor	int(11)
director	int(11)
running_time	int(11)
cost	int(11)
takings	int(11)

movie_people

Field	Type
id	int(11)
full_name	varchar(255)
is_actor	tinyint(1)
is_director	tinyint(1)

movie_type

Field	Type
id	int(11)
label	varchar(100)

Movie Data

movie_main

id	name	type	year	lead_actor	director	running_time	cost	takings
1	Bruce Almighty	5	2003	1	2	102	10	15
2	Office Space	5	1999	5	6	90	3	90
3	Grand Canyon	2	1991	4	3	134	15	10

movie_people

id	full_name	is_actor	is_director
1	Jim Carrey	1	0
2	Tom Shadyac	0	1
3	Lawrence Kasdan	0	1
4	Kevin Kline	1	0
5	Ron Livingston	1	0
6	Mike Judge	0	1

movie_type

id	label
1	Sci Fi
2	Drama
3	Adventure
4	War
5	Comedy
6	Horror
7	Action
8	Kids