CENTER FOR TEXTUAL STUDIES AND
DIGITAL HUMANITIES

DIGH 402 - Introduction to Digital Humanities Design and Programming

Spring Semester 2015

Week 2

Databases - MySQL

creating DBs and tables:

- names (formally known as *identifiers*)
    - should be clear, meaningful, and ‘easy to type’ in queries etc
    - should only contain letters, numbers, and underscore (no spaces)
    - should not be the same as an existing keyword (eg: SQL term etc)
    - column names are case-insensitive (often best to use lowercase throughout)
    - unique with DB scope (DB realm)
        eg: a table cannot have two columns with the same name
            a DB cannot have two tables with the same name

to name DB data:

- determine the DB name
- determine the table name
- determine the column name in the table

Databases - MySQL

column types:

- determine data type for a column
    - needs to be explicitly specified
    - specifies type of data that can be stored in the column
- three primary types
    - Text (strings..)
    - Numbers
    - Dates and Times
- each primary type has a number of variants
    - Text (CHAR, VARCHAR, TEXT…)
    - Numbers (INT, FLOAT, DOUBLE...)
    - Dates and TImes (DATE, DATETIME, TIMESTAMP...)
- many types can take an optional ‘length’
    - eg: INT(10), VARCHAR(150)
    - excessive length will be truncated
    - lengths for numeric types determines output & not input length
- TIMESTAMP automatically set to current date and time
- other variants are available such as BLOB...

Databases - MYSQL

Choosing a column type

- decide whether column should store text, number, or date/time

    - often an obvious decision
    - date/time can be stored either specifically as a date or as text or as a number
        - further manipulation and use dependent on type

- select subtype for data per column

    - userid = int (10)
    - username = varchar(200)
    - usercreated = timestamp

- CHAR or VARCHAR?
    - CHAR always uses specified full string length
    - VARCHAR only uses space required for stored data
    - speed of fixed size CHAR vs flexibility of VARCHAR

Databases - MYSQL

Choosing any other column properties

- every column can be set to NOT NULL
    - NULL = no value

- default value for any column can also be set
    - if no default value, and no value set, NULL will be set for a new record
    - NOT NULL set, and no value set, error will occur for a new record

- UNSIGNED limits the stored data to positive numbers and zero
    - used with number types

- ZEROFILL pads extra space with zeros
    - used with number types
    - automatically UNSIGNED

Databases - MYSQL

Finishing your columns

- define and identify your PRIMARY KEY
    - nearly always a numerical value & must always have a value
    - unique way to refer to a given record
        eg: userid
    - can be defined arbitrarily
    - critical for referencing and finding a record

- identify those columns that cannot store a NULL value

- always specify a numeric type as UNSIGNED if a negative is either not necessary or illegal

- if applicable establish the default value for a column

- check and double check that your table can store the required data

Databases - MYSQL

Finishing your columns

- INDEX basically maintains a record of the values in a specified column of a given table
    - allows FULLTEXT search etc (MyISAM)
    - improves performance when retrieving records
    - slightly reduces performance when inserting or updating records

- KEY (Primary and Foreign)
    - each table should have one primary key
    - primary key in one table often linked as a foreign key in another
        - foreign key constraints, cascading updates and deletes... (InnoDB)

- AUTO_INCREMENT
    - added to a given column
    - automatically increments the column upon new insert by next highest number
    - eg: 1, 2, 3, 4 etc

etc, etc...

Databases - SQL

Example Queries

```
SELECT FirstName from Employees
SELECT FirstName from Employees WHERE FirstName = 'Emma'

INSERT INTO Employees VALUES ('Emma', 'Smith', '773-749-9246')

DELETE FROM Employees WHERE LastName = 'Smith'

UPDATE Employees SET PhoneNumber = '779-751-9248'  WHERE LastName = 'Smith'
UPDATE Employees SET PhoneNumber = '779-751-9248'  WHERE LastName = 'Smith'
AND FirstName = 'Emma'
UPDATE Employees SET PhoneNumber = '779-751-9248'  WHERE id = 223
```

- DISTINCT

```
SELECT DISTINCT FirstName from Employees
```

- SQL is not case sensitive
- NB: this is not an exhaustive list!

Databases - SQL & PHPMyAdmin

Development processes…

- PHP front-end to manage and control installed MYSQL
    - create new database
    - populate DB from scratch or import existing DB
    - create tables
    - add data per table and field
    - perform batch operations on a selected DB
    - manage users and privileges per DB
    - export DB or table for backup...

and on and on...

- effectively provides a front-end, graphical solution for MYSQL management
    - terminal/command line is standard interface for MYSQL management without a 3rd party tool such as PHPMyAdmin

Databases - PHPMYADMIN

Setup

- LAMP, XAMPP, WAMP - Download
  - OS X can use Homebrew to install PHP, MariaDB, and use built-in Apache 2
  - http://vanbosse.be/blog/detail/a-homebrew-lamp-stack

- XAMPP or Homebrew? Your choice.

- Raspberry Pi Setup

- add project directory, add HTML, CSS, JS, PHP etc files

- load phpmyadmin and setup mysql database

Databases - PHPMYADMIN

Create database and table using PHPMYADMIN - USERS

- open PHPMYADMIN
    - http://localhost/phpmyadmin/

- create new database with UTF-8 Unicode collation and name ‘402framework’ & storage engine will be MyISAM
    - UTF-8mb4 offers even broader unicode support (at a size cost…)

- create a new table and name ‘users’
    - add 5 columns to your new table
    - add the following column names
        userid, username, firstname, lastname, usercreated

Databases - PHPMYADMIN

Create database and table using PHPMYADMIN - USERS

	Type	Collation	Attributes	Null	Default	Extra	Primary
userid	int(10)		unsigned	No	None	auto_increment	Yes
username	varchar(30)	utf8_unicode_ci		No	None		
firstname	varchar(30)	“ ”		No	None		
lastname	varchar(50)	“ “		No	None		
usercreated	timestamp	“ “		No	CURRENT_TIMESTAMP		

Databases - PHPMYADMIN

Create database and table - USERS

- CREATE DATABASE 402framework;

- USE 402framework;

- CREATE TABLE users {
    userid INT UNSIGNED NOT NULL AUTO_INCREMENT,
    username VARCHAR(30) NOT NULL,
    firstname VARCHAR(30) NOT NULL,
    lastname VARCHAR(50) NOT NULL,
    usercreated TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,

    PRIMARY KEY (user_id)

}

- SHOW TABLES;

- SHOW COLUMNS FROM users;

Databases - PHPMYADMIN

INSERT INTO database and table - USERS

- INSERT INTO users (userid, username, firstname, lastname, usercreated) VALUES(NULL, ‘user1’, ‘yvaine’, ‘smith’, NOW());

OR

INSERT INTO users (username, firstname, lastname) VALUES(‘user1’, ‘yvaine’, ‘smith’);

NB: NOW() is a function that returns the current date and time

Databases - PHPMYADMIN

UPDATE database and table - USERS

- UPDATE users SET username='tristanwall' WHERE userid=6;

Databases - PHPMYADMIN

SELECT FROM database and table - USERS

- SELECT * FROM users;

PHP Example

MySQL Database

- create a database called ‘402framework’

- add a table called ‘users’ using the outlined properties

- add some data to this new table

Databases - PHPMYADMIN

Example SELECT Queries

- SELECT * FROM users;
- SELECT username, usercreated FROM users;
- SELECT * FROM users WHERE userid=3;
- SELECT username, usercreated FROM users WHERE userid=3;
- SELECT userid FROM users WHERE username=”yvaine”;
- SELECT DISTINCT username FROM users;
- SELECT userid FROM users WHERE firstname=”yvaine” AND lastname=”wall”;
- SELECT userid FROM users WHERE firstname=”yvaine” OR lastname=”wall”;
- SELECT * FROM users WHERE username LIKE 'yvaine%';
- SELECT * FROM users WHERE username LIKE '%yvaine';
- SELECT * FROM users WHERE lastname IN ('issit', 'stormhold');
- SELECT * FROM users WHERE userid BETWEEN 2 AND 5;
    - SELECT * FROM users WHERE userid NOT BETWEEN 2 AND 5;
    - SELECT * FROM users WHERE lastname BETWEEN 'd' and 'm';
    - SELECT * FROM users WHERE lastname BETWEEN 'd' and 'm' ORDER BY firstname;

...

PHP and MySQL

Expand 402framework database

- add new tables for
    - content, content_type, content_lookup

- consider each table as separate data except ‘content_lookup’

- consider required table properties for ‘content’ and ‘content_type’
    - data type, collation, attributes, null, default, and any extras and indices

- content_lookup properties
    - artificially replicating foreign keys in MyISAM
    - primary keys
    - combine ‘content’, ‘content_type’, and ‘users’

- why an artificial ‘lookup’ table?
    - speed
    - easier to read and reference
    - often personal preference

- alternatives such as
    - embed reference ID in ‘content’ table

http://sqlfiddle.com/#!2/58892/20

Databases - PHPMYADMIN

Create database and table using PHPMYADMIN - CONTENT

	Type	Collation	Attributes	Null	Default	Extra	Primary
contentid	int(10)		unsigned	No	None	auto_increment	Yes
contentname	varchar(150)	utf8_unicode_ci		No	None		
contentdesc	text	“ ”		No	None		
contenttext	text	“ “		No	None	fulltext	
contentcreated	timestamp	“ “		No	CURRENT_TIMESTAMP		

Databases - PHPMYADMIN

Create database and table using PHPMYADMIN - CONTENT_TYPE

	Type	Collation	Attributes	Null	Default	Extra	Primary
content_type_id	int(10)		unsigned	No	None	auto_increment	Yes
content_type_name	varchar(150)	utf8_unicode_ci		No	None		
content_type_desc	text	“ ”		No	None		

Databases - PHPMYADMIN

Create database and table using PHPMYADMIN - CONTENT_TYPE_LOOKUP

	Type	Collation	Attributes	Null	Default	Extra	Primary
content_id	int(10)		unsigned	No	None		Yes
content_type_id	int(10)		unsigned	No	None		Yes
user_id	int(10)		unsigned	No	None		Yes

Databases - PHPMYADMIN

Using our new tables and DB structure

- manage, list, and provide work records for our users, such as
    - who is a user of the framework
    - content created and when
    - content types created per user (and again when)

- potential patterns?

- use the ‘content_type_lookup’ table to ask the following queries
    - current content by content_type
    - current content by user/users
    - current content by timestamp …

- how would this change if we added the following columns to our ‘users’ table
    - gender
    - age
    - country

and so on...

Databases - PHPMYADMIN

Selecting data from our new tables - queries against ‘content_type_lookup’

- find total number of content items of a given content type

```
SELECT COUNT(content_lookup.content_type_id) FROM content_lookup WHERE content_type_id=1;
```

- find all content with content name, content id, and username by user id

```
SELECT content.contentid, content.contentname, users.username FROM content_lookup, content, users WHERE content_lookup.content_id=content.contentid AND users.userid=content_lookup.user_id AND content_lookup.user_id=1
```

- find all content with content type name and content name by content type id

```
SELECT content_type.content_type_name, content.contentname FROM content, content_type, content_lookup WHERE content_lookup.content_id=content.contentid AND content_lookup.content_type_id=content_type.content_type_id AND content_lookup.content_type_id=1;
```

and many more...

PHP and MySQL

SELECT FROM database and table - USERS

- SELECT * FROM users;

PHP Example

- add user privileges to 402framework database
  - admin user & basic query user
  - different privileges relative to user requirements and framework security

PHP and MySQL

Basic PHP MySQL connection - admin or basic user privileges

```
<?php
$con = mysqli_connect('localhost', 'username', 'password', '402framework');
if (!$con) {
    die('Could not connect: ' . mysql_error());
}

$result = mysqli_query($con,"SELECT * FROM users");

while($row = mysqli_fetch_array($result))
  {
  echo 'user = '.$row['username'] . " & created = " . $row['usercreated'];
  echo "<br>";
  }

mysqli_close($con);
?>
```

PHP and MySQL

Test basic PHP MySQL connection & SELECT queries - against USERS table

- we can modify basic connection to query ‘users’ table
    - SELECT username, usercreated FROM users;
    - SELECT * FROM users WHERE userid=3;
    - SELECT DISTINCT username FROM users;

- output results in a tabular format

    PHP Example

PHP and MySQL

Test basic PHP MySQL connection & SELECT queries - tabular output

```
<?php
$con = mysqli_connect('localhost', 'username', 'password', '402framework');
if (!$con) {
    die('Could not connect: ' . mysql_error());
}

$result = mysqli_query($con,"SELECT * FROM users");

echo '<table>';
echo '<th>username</th>';
echo '<th>created</th>';

while($row = mysqli_fetch_array($result))
  {
  echo '<tr>';
  echo '<td>'.$row['username'].'</td>';
  echo '<td>'.$row['usercreated'].'</td>';
  echo '</tr>';
  }

echo '</table>';

mysqli_close($con);
?>
```

PHP and MySQL

Set user privileges

- create 2 new users
  - one as an ‘admin’ account
  - another as a ‘user’ account
- ‘admin’ user should have all privileges set for the specified database, eg: 402framework
- ‘user’ only needs the following privileges for the specified database at this moment
  - Data
    - SELECT