MySQL and PHP

CS/IT 490 WD, Fall 2013

Last update 2013-10-29

Written by Rachel J. Morris

Breakdown

- Table Modification
  - Create Table
  - Alter Table
  - Drop Table
- Data Modification
  - Insert
  - Update
  - Delete
  - Select

Breakdown

- Data Modification (continued)
  - Select … Where
- Handy things
  - Aliases
  - Inner Joins
- Relationships
  - 1:N
  - N:N

Breakdown

- PHP & MySQL
  - Connecting to a Database
  - Sending Queries
  - Retrieving Results

Table Modification Statements

CREATE TABLE

```
CREATE TABLE table_name (
        column1 datatype1 properties,
        column2 datatype2 properties,
        …
        columnn datatypen properties
);
```

Table Modification Statements

CREATE TABLE

Common Data Types

VARCHAR(size)	A string, must specify size
INT	An integer
BOOLEAN	A boolean value `TRUE, FALSE – aliases for 1 and 0`
DATETIME	The date & time, in the format YYYY-MM-DD HH:MM:SS `2013-10-17 13:02:00`
FLOAT	A float

Table Modification Statements

CREATE TABLE

Properties

NOT NULL	This field CAN NOT be left empty and will cause an error if not included on insert.
AUTO_INCREMENT	If this field is not specified on insert, a new value will be added, which is incremented for each field. Good for primary keys.
DEFAULT	Set a default value when none is provided on insert. user_name VARCHAR(100) DEFAULT 'unset'
UNIQUE	The value in a record in this column can only appear once.
FLOAT	A float

Data Modification Statements

CREATE TABLE

Set one column primary key

```
PRIMARY KEY ( column_name )
```

Set multiple columns as a primary key

```
PRIMARY KEY ( column_name1, column_name2, ... )
```

Set a foreign key

```
FOREIGN KEY ( column_name1 )
REFERENCES Table2 ( column_name2 )
```

Table Modification Statements

ALTER TABLE

Add a column:

```
ALTER TABLE table_name
ADD column_name data_type
```

Remove a column:

```
ALTER TABLE table_name
DROP column_name
```

Table Modification Statements

You can rename a table with

```
RENAME TABLE oldname TO newname
```

Table Modification Statements

DROP TABLE

```
DROP TABLE table_name
```

Data Modification Statements

INSERT Data

```
INSERT INTO table_name
( column1, column2, …, columnn )
VALUES
( value1, value2, …, valuen )
```

You can also just specify values, but they must be in the same order as the columns are in the table.

Make sure to enclose strings and dates in single quotes ' … '

```
INSERT INTO Employee
( first_name, last_name, start_date )
VALUES
( 'Elaine', 'Marley', CURDATE() )
```

Data Modification Statements

UPDATE Data

```
UPDATE table_name
SET column1 = value1, column2 = value2, …, columnn = valuen
WHERE column = value
```

Example:

```
UPDATE Gamers
SET is_active = FALSE
WHERE last_play_date < '2013-01-01'
```

Data Modification Statements

DELETE Data

```
DELETE FROM Products
WHERE inactive = TRUE
```

Data Modification Statements

SELECT Data

Select everything from one table:

```
SELECT * FROM Table
```

Select all records, but only some columns, from one table:

```
SELECT
      column1, column2, …, columnn
FROM Table
```

Select records that meet the criteria, and only certain columns, from one table:

```
SELECT
      column1, column2, …, columnn
FROM Table
WHERE column = value
```

Data Modification Statements

SELECT Data

Select records across multiple tables:

```
SELECT
      Table1.column1, Table1.column2,
      Table2.column1, Table2.column2
FROM Table1
INNER JOIN Table2
ON Table1.table2_id = Tabel2.table2_id
```

Aliases

- You can rename a table or a column from within your query:

```
SELECT title
FROM movie AS theater_listing
WHERE type = 'theater'
```

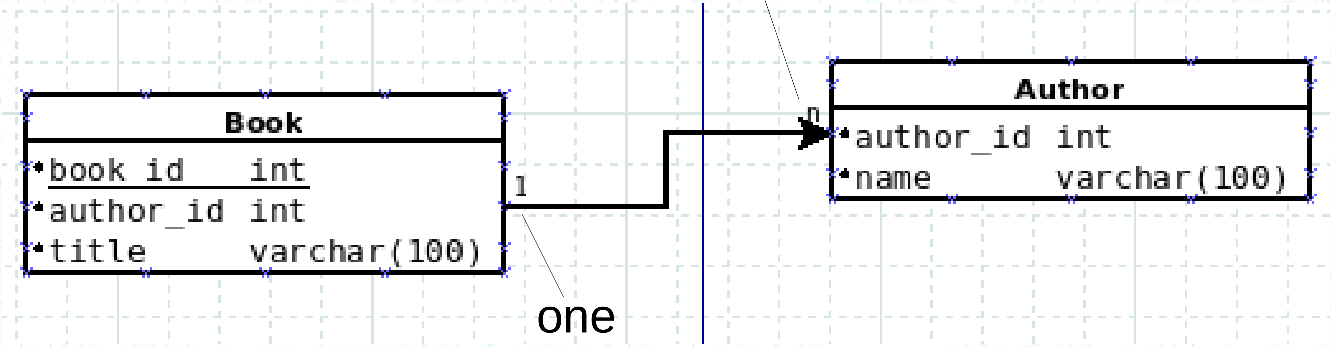
```
SELECT Publisher.name as pub_name, Game.name as game_name
FROM Publisher
INNER JOIN Game
ON Game.publisher_id = Publisher.publisher_id
```

Relationships

- You can easily specify a 1:1 or 1:N with Foreign Keys.
- One table has a foreign key that points to a second table's primary key.

Relationships

- For a very simple book database, assuming a book can only have **one** author, but an author can have **many** books:



```
Book
book_id    int
author_id  int
title      varchar(100)
```

```
Author
author_id  int
name       varchar(100)
```

n = many

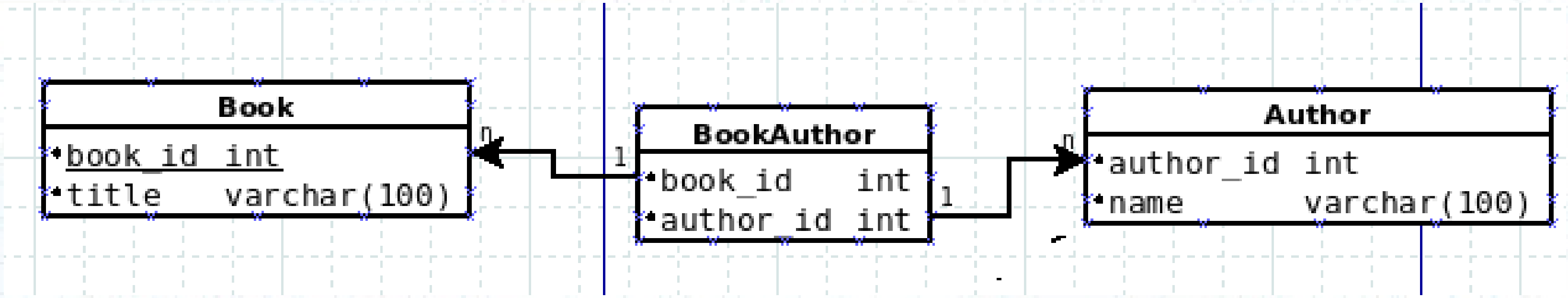
one

Relationships

- For two tables to have an N:N (many-to-many) relationship, we need an intermediate table.
- A book may have **many** authors, and an author may have **many** books.
- We need a “Book-Author” intermediate table to link them together.

Written by Rachel J. Morris

Relationships



Book

```
book_id  int
title    varchar(100)
```

BookAuthor

```
book_id     int
author_id   int
```

Author

```
author_id  int
name       varchar(100)
```

Each record links an author to a book.
With this, we can give a book multiple
authors, and an author multiple books.

book_id	title
1	Guide to Cheese
2	How to write l33t codez
3	How to take over the world

book_id	author_id
1	1
1	2
1	3
2	1
3	1

author_id	name
1	Bob Osm
2	Olivia Jerden
3	Teslen

PHP & MySQL

- Note that the library for using MySQL in PHP has changed within the past few years.
- You should be using the PDO class now.

PHP & MySQL

Connecting to a host

```
// First, you need to know a hostname, database name,
// username, and password.
$hostname = '';
$database = '';
$username = '';
$password = '';
```

```
// Then, you build a connection string, which contains the
// Hostname & Database name:
$connection = "mysql:host=" . $hostname . ";" . "dbname=" . $database;
```

PHP & MySQL

Connecting to a host

```
// Then, enclosed in a Try/Catch block we try to create
// a new instantiation of the PDO object.

try
{
    $dbHandler = new PDO(
        $connection,
        $username,
        $password
    );
}
catch( PDOException $error )
{
    // Display error message on failure
    print_r( $error->getMessage() );
}
```

The connection string from the previous slide.

PHP & MySQL

Connecting to a host

```
$dbHandler
```

- If the creation of the new PDO item was successful, then you can now send in queries to the $dbHandler instance.

PHP & MySQL

Submitting a Query

- A SELECT statement and other types of statements are handled with different functions.
- The main difference is that SELECT will return data from the database.
- INSERT, UPDATE, CREATE, etc. will only return the amount of rows that have been affected.

PHP & MySQL

Submitting a Query

- A SELECT statement:

```
$query = "SELECT * FROM Autor";
$rows = array();
foreach( $dbHandler->query( $query ) as $row )
{
    array_push( $rows, $row );
}
```

- Other statement types:

```
$query = "INSERT INTO Author";
$query .= " ( name, email_address, homepage )";
$query .= " VALUES ";
$query .= " ( 'Bob sampleton', 'bob@samplebooks.info', 'samplebooks.info' )";

$result = $dbHandler->exec( $query );
```

PHP & MySQL

Submitting a Query

- A SELECT statement:

```
$query = "SELECT * FROM Autor";
$rows = array();
foreach( $dbHandler->query( $query ) as $row )
{
    array_push( $rows, $row );
}
```

- Notice that, similar to reading a text file, we need to use a loop in order to read the return results of the query, one record (or row) at a time.
- Note that here, we've created a “$rows” array, and are storing all records in this array.

Variables in your Query

- Remember that the period . Is used to concatenate strings together in PHP.
- It might be useful to write your query out on multiple lines with the .= operator

```
$query = "INSERT INTO Food";
$query .= "( name, calories )";
$query .= " VALUES ";
$query .= "( '$info[name]', $info[calories] )";
```

- Notice that the $info[name] variable is enclosed in single-quotes, while $info[calories] is not.
- This is because the name column is for strings (must have single quotes surrounding), and the calories column is for integers.

Variables in your Query

- You might also want to include variables in your queries.
- As shown last slide, you can just reference a variable name from within a string, and it will output.
- For an associative array, you don't need to enclose the key in quotes.

References

-