Lecture Notes 9
Database Connectivity from Python and PHP Programs

Anand Tripathi

CSci 4131

Internet Programming

Topics

- Basic Concepts in relational databases
  - Tables, Joins
- SQL queries and database operations
- Python and PHP functions for database access

- Please read Chapter 18 of Deitel's book

Relational Database Model

- Data is organized as table -- rows and columns .
- Each row in a table represents some related set of data items.
- Primary key: One of the column values is used for indexing in the table.
  - This value is unique for each row.
- A database may contain multiple tables, with some values common in different tables.
  - Relation among different tables
- SQL (Structured Query Language) is used to query or update the tables.

Example: Authors Table

authorID	firstName	lastName
1	Harvey	Deitel
2	Paul	Deitel
3	Tem	Nieto
4	Kate	Steinbuhler
5	Sean	Santry
6	Ted	Lin
7	Praveen	Sadhu
8	David	McPhie
9	Cheryl	Yaeger
10	Marina	Zlatkina
11	Ben	Wiedermann
12	Jonathan	Liperi

Example: Authors Table

Three fields are defined:

authorID: This is the primary key, defined as Integer;
auto-increment assigns next integer value for this field whenever a new row is added.This field value has to be unique for each row.

firstName: a String containing author's first name

lastName: a String containing author's last name

Example in Chapter 18

This example defines 4 tables:

- **Authors table**
- **Publishers table**
- **AuthorISBN table**
- **Titles table**

Publishers table

This table contains two fields:

publisherID: The publisher's ID number stored as an auto-increment Integer. It is the primary key.

publisherName: Name stored as String

publisherID	publisherName
1	Prentice Hall
2	Prentice Hall PTG

AuthorISBN table

isbn String containing the ISBN of a book

authorID ID number of one of the authors of a book whose ISBN is stored in the row.

Here these two fields together form a unique value and therefore they form composite primary key.

AuthorISBN table

isbn	authorID
0130125075	1
0130125075	2
0130161438	1
0130161438	2
0130161438	3
0130284173	3
0130284173	6
0130284173	7
0130284181	8

Titles table

isbn String

title String

editionNumber String

description String

publisherID Integer; must correspond to ID in the Publishers table

Copyright Year of copyright -Integer

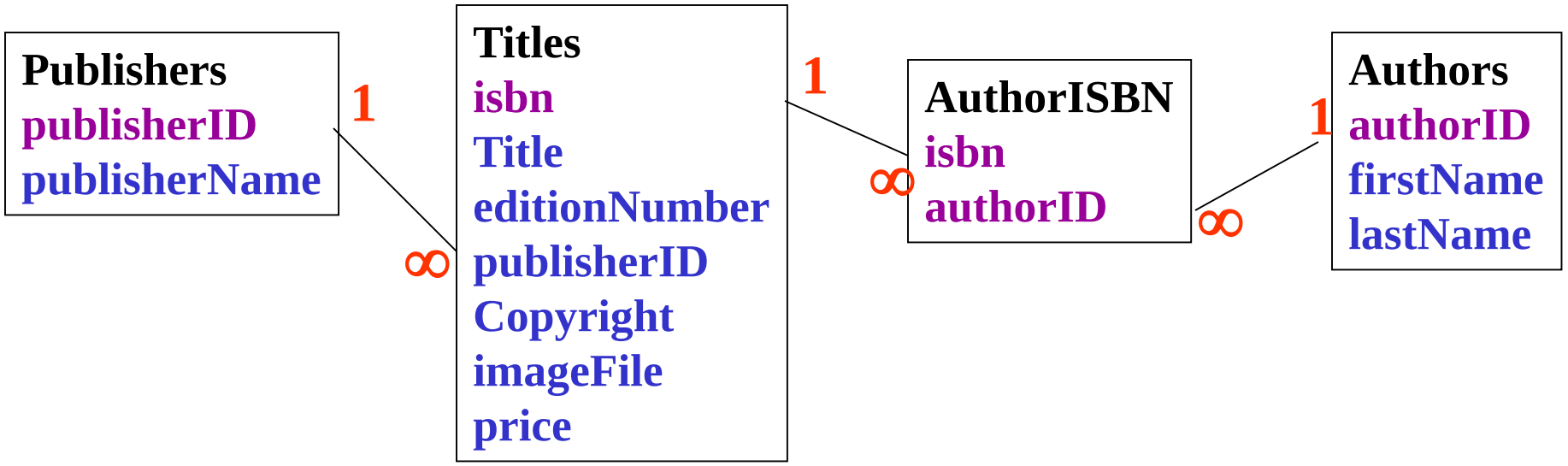
imageFile String for file-path name for book's cover image

Price Real number

One row of the Titles table

isbn	0130125075
title	Java How to Program (Java 2)3Learn to program in the Java programming language.
description	This edition has been updated to the Sun Microsystems Java 2 platform (formerly called Java 1.2). The book includes the Swing GUI components and new chapters on JDBC, Servlets, RMI, Collections and JavaBeans.
editionNumber	1
copyright	2000
imageFile	jhtp3.jpg
price	$69.95

Table Relationships



SQL (Structured Query Language)

- **Query or update the database**
  - **Read a table's rows matching some criteria**
  - **Select only a subset of rows**
  - **Make queries spanning multiple tables**
    - **Join operation**
- **Update the database**
  - **Insert data in a table**
  - **Update data in a table**
- **Create a new table**

SQL command keywords

- CREATE Creates a new table
- SELECT Select (retrieve) columns form one or more tables
  - FROM Specifies tables
  - WHERE Selection criteria for rows
  - INNER JOIN Join rows from multiple tables
  - ORDER BY Ordering criteria for rows
- INSERT Insert data in a specified table
- UPDATE Updates data in a specified table
- DELETE Deletes data from a specified table
- DROP Delete an existing table
- COUNT Count the number of records that satisfy a given search criteria

Create a new Table

CREATE TABLE Authors (authorID INTEGER AUTO_INCREMENT, firstName CHAR(20), lastName CHAR(20), PRIMARY KEY(authorID));

create table Publishers (publisherID INTEGER AUTO_INCREMENT, publisherName Char(20), PRIMARY KEY(publisherID));

CREATE TABLE AuthorISBN (authorID INTEGER, isbn INTEGER);

CREATE TABLE Titles (isbn CHAR(20), title CHAR(100), editionNumber CHAR(10), copyright INTEGER, description BLOB, publisherID INTEGER, imageFile CHAR(255), price REAL, unique (isbn));

INSERT Operation

INSERT INTO tableName (columnName1, columnName2,…columnNameN)

VALUES (value1, value2, … valueN)

INSERT INTO Authors (firstName, lastName)

VALUES ('Jack', 'Jumping')

Examples of insert

INSERT INTO Authors (authorID, firstName, lastName) VALUES (null, 'Harvey', 'Deitel');
INSERT INTO Authors (authorID, firstName, lastName) VALUES (null, 'Paul', 'Deitel');

insert Publishers (publisherID, publisherName) VALUES(null, 'Prentice Hall');
insert Publishers (publisherID, publisherName) VALUES(null, 'Prentice Hall PTG');

INSERT INTO Titles (isbn, title, editionNumber, copyright, description, publisherID, imageFile, price)
VALUES(0130923613, 'Python How to Program', '1', 2002, 'Something about Python', 1, 'python.img', 69.99);

UPDATE Operation

```
UPDATE  tableName
SET columnName1 = value1,
        columnName2 = value2,
            …
        columnNameN = valueN
WHERE criteria
```

Example:

```
UPDATE  Authors
SET lastName = 'Jones',
WHERE lastName = 'Jumping' AND firstName = 'Jack'
```

INSERT INTO Authors (firstName) VALUES ('Jack')

select * from Authors where lastName IS NULL;

SELECT Query

SELECT * FROM tablename

Select all columns from the given table.

SELECT * FROM Authors

SELECT authorID, lastName FROM Authors

WHERE Query

***WHERE* clause is used to specify selection criteria.**

SELECT columnName1, columnName2,…

FROM tableName

WHERE criteria

Example:

SELECT title, editionNumber, copyright

FROM Titles

WHERE copyright> 1999

Pattern matching can be specified in the WHERE clause using LIKE operator.

Pattern Matching in WHERE

Pattern matching can be specified in the WHERE clause using LIKE operator.

Example:

```
SELECT authorID, firstName, lastName
FROM Authors
WHERE lastName LIKE 'D%'
```

Pattern Matching in Oracle

%	**is equivalent to *, any character**
_	**means any character**
[a-d]	**any single character in given range**
[abxyz]	**any character in the given set**
[^a-d]	**any character not in the given set or range**

SQL> select title from Titles where title LIKE '_____How%';

Java How to Program

Pattern Matching in Oracle

SQL> select title from Titles where title LIKE '%How%';

TITLE

Python How to Program

C# How to Program

Java How to Program

Internet and WWW How to Program

TITLE

ORDER By Clause

SELECT columnName1, columnName2,…

FROM tableName

ORDER BY column ASC

ASC for ascending order,Use DESC for descending order,

Example:

SELECT authorID, firstName, lastName

FROM Authors

ORDER BY lastName ASC

ORDER By Clause

Example:

```
SELECT isbn, title, editionNumber, copyright, price
FROM Titles
WHERE title LIKE '%How to Program%'
ORDER BY title ASC, copyright DESC
```

JOIN Operation: Merging Data from Multiple Tables

```
SELECT columnName1, columnName2,…
FROM tableName
INNER JOIN table2
ON  table1.columnName = table2.columnName
```

Example:

```
SELECT firstName, lastName, isbn
FROM Authors
INNER JOIN AuthorISBN
   ON Authors.authorID = AuthorISBN.authorID
ORDER BY lastname, firstName
```

JOIN Operation: using Oracle

Example:

Select firstName, lastName, isbn

From Authors, AuthorISBN

Where Authors.authorID = AuthorISBN.authorID;

Database Connection from Python

Click here to see this example

See online example on course-page under Python/DatabaseConnection

```
#!/usr/bin/python
import cgi
import cgitb; cgitb.enable()
import MySQLdb

print 'content-type: text/html\n'
print '<html><body>'
form = cgi.FieldStorage()
tableSelection = form.getvalue('tableSelection')
query = 'select * from ' +  tableSelection
#connect to MySQL
db = MySQLdb.connect(host="egon.cs.umn.edu", user="C4131S14U1",
                passwd="9876", port=3307)
```

Database Connection from Python

```
#select database
db.select_db("C4131S14U1")
#create cursor object, let you execute query
cursor = db.cursor()
#query database
cursor.execute(query)
#parse query results
print '<table border = "1" cellpadding = "3" cellspacing = "2" style = "background-color:
#ADD8E6">'
for row in cursor:
    print '<tr>'
    for col in row:
        print '<td>' + str(col) + '</td>'
    print '</tr>'
print '</table>'
print '</body></html>'
```

Cookies

- HTTP Protocol is stateless.
- To establish a relationship between a client's current request with any of the previous requests, cookies are used.
- Cookies are name=value pair of strings.
- Cookies are sent from the server to client. The client resends the cookies back to the server in any of the subsequent requests.
- See RFC 2109 and RFC 2965

Set-Cookie in Response Header

- Set-Cookie: *NAME*=*VALUE*; expires=*DATE*; path=*PATH*; domain=*DOMAIN_NAME*; secure

- This information is stored by the browser.

- *NAME*=*VALUE*

  This string is a sequence of characters excluding semi-colon, comma and white space. If there is a need to place such data in the name or value, some encoding method such as URL style %XX encoding is recommended, though no encoding is defined or required. This is the only required attribute on the Set-Cookie header.

Setting Cookies in Python CGI Programs

See example on course webpage under Python/Cookies

- A CGI program can set cookies with the output that it sends to a client.
- Following example is a Python version of on how to set cookies.
  - A form is present to the user using which the user will ask for certain cookies to be set. (ccokies.html)
  - On form submission, the CGI program generates and sends cookies back to the user's browser. (writecookies.cgi)
  - When user sends to request to the CGI program readcookies.cgi, it echoes back the cookies sent by the user's browser.
- See this program here

Cookies.html

```
<form method = "post" action = "writecookies.cgi“    style = "font-size: 10pt">

        <strong>Name:</strong><br />
        <input type = "text" name = "NAME" /><br />

        <strong>Height:</strong><br />
        <input type = "text" name = "HEIGHT" /><br />

         <strong>Favorite Color:</strong><br />
        <input type = "text" name = "COLOR" /><br />

         <strong>Cookie age:</strong><br />
        <input type = "text" name = "TIME" /><br />

        <input type = "submit" value = "Write Cookie"
         style = "background-color: #F0E86C; color: navy;    font-weight: bold" /></p>
 </form>
```

writecookies.cgi

```
#!/usr/bin/python
import cgi
import cgitb; cgitb.enable()
import Cookie
import time

cookie = Cookie.SimpleCookie()
form = cgi.FieldStorage()
name = form.getvalue('NAME') .value
height = form['HEIGHT'].value
color = form['COLOR'].value
timeout = int(form['TIME'].value.strip())

cookie['NAME'] = name
cookie['HEIGHT'] = height
cookie['COLOR'] = color
```

writecookies.cgi

```
# values expires in timeout seconds
cookie['NAME']['max-age'] = timeout
cookie['HEIGHT']['max-age'] = timeout
cookie['COLOR']['max-age'] = timeout

print cookie
print 'content-type: text/html\n'
print '<html><body>'
print 'Cookies successfully written. Click <a href="readcookies.php">here</a> to view.'
print '</body></html>'
```

readcookies.cgi

```
#!/usr/bin/env python
import os
import cgi
import cgitb; cgitb.enable()
import Cookie
import time
if 'HTTP_COOKIE' in os.environ:
  cookies = os.environ['HTTP_COOKIE']
  cookies = cookies.split('; ')
handler = {}
for cookie in cookies:
   cookie = cookie.split('=')
   handler[cookie[0]] = cookie[1]
```

readcookies.cgi

```
print 'content-type: text/html\n'
print
print '<html><body>'
print 'Cookies sent by browser.'
print "</br>"
for key,value in handler.iteritems():
    print key, "=", value
    print "</br>"
print '</body></html>'
```

Your MySQL Account

- Information was emailed to you on October 14.
- Login into your account and change password immediately

- module load soft/mysql
- mysql -u(mysqluser) -hegon.cs.umn.edu -P3307 -p (databasename)

- databasename is the same as mysqluser

readcookies.php

```
<table border = "5" cellspacing = "0" cellpadding = "10">
      <?php

        // iterate through array $_COOKIE and print
        // name and value of each cookie
        foreach ( $_COOKIE as $key => $value )
          print( "<tr>
            <td bgcolor=\"#F0E68C\">$key</td>
            <td bgcolor=\"#FFA500\">$value</td>
            </tr>" );
      ?>

    </table>
```

Example of Connecting to Database from PHP

Click here to see this example.

- A form is displayed to choose a table to display.
- This form is submitted to databasew.php, a CGI program.
- This CGI program will query a MySQL database belonging to the instructor for the specified table.
- Based on the response data returned by the MySQL server, it will generate an HTML document containing data in a tabular form.

Example of Connecting to Database from PHP

Click here to see this example.

```
<!– File  data.html     -->
<!-- Querying a MySQL Database -->

<html xmlns = "http://www.w3.org/1999/xhtml">
  <head>
    <title>Sample Database Query</title>
  </head>

  <body style = "background-color: #F0E68C">
    <h2 style = "font-family: arial color: blue">
      Querying a MySQL database.
    </h2>
```

```
<form method = "post" action = "database1.php">
        <p>Select a field to display:

          <!-- add a select box containing options -->
          <!-- for SELECT query                    -->
          <select name = “tableSelection">
            <option selected = "selected">*</option>
            <option>ID</option>
            <option>Title</option>
            <option>Category</option>
            <option>ISBN</option>
          </select>
        </p>

        <input type = "submit" value = "Send Query"
          style = "background-color: blue;
          color: yellow; font-weight: bold" />
      </form>
   </body>
</html>
```

PART of database1.php file:

```
<html xmlns = "http://www.w3.org/1999/xhtml">
   <head>
      <title>Search Results</title>
   </head>

   <body style = "font-family: arial, sans-serif"
      style = "background-color: #F0E68C">

    <?php
         extract( $_POST );

         // build SELECT query
        $query = "SELECT * from $tableSelection";
```

```
// Connect to MySQL
• if ( !( $database = mysql_connect( “db.cselabs.umn.edu:3307",
        “C4131S12U1", "password" ) ) )`
        die( "Could not connect to database" );

    // open Products database
    if ( !mysql_select_db( “C4131S14U1", $database ) )
        die( "Could not open 4131Examples database" );

    // query Products database
    if ( !( $result = mysql_query( $query, $database ) ) ) {
        print( "Could not execute query! <br />" );
        die( mysql_error() );
    }
?>
<h3 style = "color: blue">
Search Results</h3>
```

```
<table border = "1" cellpadding = "3" cellspacing = "2"
        style = "background-color: #ADD8E6">
        <?php
          // fetch each record in result set
          for ( $counter = 0;  $row = mysql_fetch_row( $result ); $counter++)
          {
             // print a table row to display results
             print( "<tr>" );
                 foreach ( $row as $key => $value )
                             print( "<td>$value</td>" );
             print( "</tr>" );
          }
          mysql_close( $database );
        ?>
 </table>
   <br />Your search yielded <strong>
      <?php print( "$counter" ) ?> results.<br /><br /></strong>
</body> </html>
```