PHP PostGreSQL Database Commands

PHP Functions that Work With PostGreSQL

- PHP has the ability to connect to many types of Database software
- most popularly MySQL, but also PostGreSQL (both shareware)
- This course is using PostGreSQL as it is a more powerful tool than MySQL (closer to Oracle/SQL Server than MS Access)
- The Good news: very limited number of PHP functions are required
- Note: all of the functions shown have to exist in PHP `<?php ?>` tags

Basic DB Processing Sequence

- In order to access information from any database, you must always perform the following steps:
  - first one has to connect to the database
  - Then run SQL commands against tables in the database
  - Process any information that is returned from the database (if there is any)
- To complete all of this for this course we only need to use the following PHP provided functions:
  - `pg_connect()`
  - `pg_query()`
  - `pg_num_rows()`
  - `pg_fetch_result()`

pg_connect()

- Performs the all important step of connecting to a database
- Returns a PostGreSQL connection object that is used by subsequent functions
- Example

```
$conn = pg_connect("host=127.0.0.1 dbname=DB_NAME user=USERID password=PASSWORD");
```

  - 127.0.0.1refers to the computer running the PHP page
    - NOTE: localhost should work as well, but the opentech server does not like this
  - dbname is the database you are connecting to (in our case userid_db)
  - user is the postGreSQL user you are connecting as
  - password is the user's PostGreSQL password
- $conn is a connection object that will be used below (does NOT have to be called connection)

pg_query()

- Used to run SQL statements against your database
- N.B. this includes INSERT, UPDATE and DELETE statements (not just SELECT queries)
- Will return a result set if a SELECT statement is given
- Example:

```
$sql = “SELECT title, year FROM movies”;
$results = pg_query($conn, $sql); //$conn was
                                  //created with
                                 //the pg_connect()
```

- `$results` is a result set that contains the title and year of any records in the movies table

pg_num_rows()

- Used to determine if any records were returned in a result set
- Used to decide if any further processing required
- Example:

```
$records = pg_num_rows($results); //$results was
                                   //created by
                                  //the pg_query()
```

- `$records` is the total number of records in the result, will usually be 0 or above (integer values as it is a count)
- Can throw a `-1`, if there was an error (usually occurs if you do not pass a result set argument)

pg_fetch_result()

- Used to retrieve any information contained in a result set from a SELECT statement
- Requires a result set argument, and a reference to a record and a column
- Example:

```
$title = pg_fetch_result($results, 0, 0);//$results was
                                          //created by
                                          //the pg_query()

$year = pg_fetch_result(($results, 0, 1);
```

- This is putting the first record's (0^{th}) first column (0^{th}) into a variable named $title, and the first record's second column (1^{st}) into a variable named $year

pg_fetch_result() (cont'd)

- Another syntax that you can use is the database table's field name versus the index of the field in the SQL SELECT statement. Example:

```
$title = pg_fetch_result($results, 0, "title");
                                    //$results was
                                    //created by
                                    //the pg_query()
$year = pg_fetch_result($results, 0, "year");
```

- This is putting the first record's (0th) "title" column into a variable named `$title`, and the first record's "year" column into a variable named `$year`
- This syntax sometimes makes it easier to see what the code is doing (less cryptic/confusing than a zero'ed index "array")

Realistic pg_fetch_result()

- Usually (though not always) a SQL SELECT statement will return multiple records
- The following gives a way to process them all:

```
$conn = pg_connect("host=127.0.0.1 dbname=DB_NAME
                        user=USERID password=PASSWORD");
$sql = “SELECT title, year FROM movies”;
$results = pg_query($conn, $sql);
$records = pg_num_rows($results);
for($i = 0; $i < $records; $i++){
   $title = pg_fetch_result($results, $i, “title”);
   $year = pg_fetch_result($results, $i, “year”);
   echo “<p>The movie “ . $title . “ was released in “ . $year .“</p>”;
}
```

- The for loop starts at the 0^{th} record and goes to the $(n-1)^{th}$ record displaying each column