Web System Architecture

- Three-tier architecture.
- Web servers: interact with users.
- Application servers: presentation and business logic.
- Database servers: store the data and communicate with the application servers.

Example: Facebook

- Database stores profiles, friends, comments, links, etc.
- App server implements “friending” and “defriending” by issuing query updates to the database servers.
- Web servers display info to users and capture clicks.

Database Connectivity

- Embedded SQL.
- Language-specific libraries:
  - SQL/CLI (Call-Level Interface) in C
  - JDBC in Java
  - PHP

Terminology

- Environment: the database.
- Connections: maintained by the DB server so app servers can perform queries or updates.
- Statements: queries and updates issued by the app servers.
- An environment may contain multiple connections. Statements are executed over connections.
JDBC

```java
import java.sql.*;
Class.forName(com.mysql.jdbc.Driver);
Connection myCon = DriverManager.getConnection(...);

• Environment parameters such as DB address, user name, password, and other system specific info are passed to getConnection.
```

Queries in JDBC

• Two Java classes:
  • Statements - can be assigned any string representing SQL query.
  • PreparedStatements - assigned a string representing SQL query at creation.

Creating Statements

• The Connection class has methods to create Statements and PreparedStatements.

```java
Statement s = myCon.createStatement();
PreparedStatement ps = myCon.createStatement(
    "SELECT beer, price FROM Sells WHERE bar = 'Hopleaf'");

• createStatement with no argument returns a Statement; with one argument it returns a PreparedStatement.
```

Executing SQL Statements

• In JDBC, queries and updates are treated differently.
  • Statement and PreparedStatement have methods `executeQuery` and `executeUpdate`.
    – One argument for Statements: the query or update to be executed.
    – No arguments for PreparedStatements.

Update Example

```java
s.executeUpdate(
    "INSERT INTO Sells VALUES('Hopleaf','Corona',6.5)"
); 

• Is s a statement or a prepared statement?
```

Query Example

```java
• ps is a PreparedStatement assigned the query "SELECT beer, price FROM Sells WHERE bar = 'Hopleaf'."

• executeQuery returns an object of class ResultSet.

ResultSet menu = ps.executeQuery();
```
**Accessing the ResultSet**

- An object of type ResultSet acts like a cursor.
- Method `next()` advances the cursor to the next tuple.
  - The first time `next()` is called, it gets the first tuple.
  - If there are no more tuples, `next()` returns the value false.

**Accessing Components of Tuples**

- If a ResultSet is referring to a tuple, we can get the components of that tuple by applying methods `getX(idx)`, to the ResultSet.
  - X is some type, and idx is the component number.
  - The value must have type X.

**Accessing Components**

```java
PreparedStatement ps = myCon.createStatement("SELECT beer, price FROM Sells WHERE bar = 'Hopleaf'");
ResultSet menu = ps.executeQuery();
while ( menu.next() ) {
    aBeer = menu.getString(1);
    aPrice = menu.getFloat(2);
    /* process the values */
}
```

**PHP**

- A scripting language for web development.
  - executed by the web server.
- Format: `<?php ...

```php
<?php
    echo "Hello, World!";
?>
```

**PHP Variables**

- Variable names must begin with $.
- Variables do not need to declared or typed.
- When a variable is assigned a value from a certain class, all class methods become available.

**String Values**

- Double quotes and single quotes have different interpretations in PHP.
- Single quotes mean literal string with no processing.
- Double quotes mean replace variable names inside the string with their values.
Examples

$ten = ‘ten dollars’;

$bud = ‘It costs $ten.’;
// value is ‘It costs $ten.’

$stella = “It costs $ten.”;
/* value is ‘It costs ten dollars.
*/

PHP Arrays

• Numeric and associative.
• Numeric arrays are indexed 0,1,...
• Example:
  $dm = array(‘Dave’, ‘Martin’, ‘Andrew’,
               “Alan”)
  // $dm[0] is “Dave”, $dm[1] is “Martin”,
  // and so on.

Associative Arrays

• Elements of an associative array $a are
  pairs $x => $y, where $x is a key string and
  $y is any value.
• If $x => $y is an element of $a, then $a[$x]
  is $y.

Associative Arrays Example

$mysqlParams = array(  
  ‘host’ => ‘cspp53001.cs.uchicago.edu’,
  ‘user’ => ‘zarko’,
  ‘password’ => ‘secret’,
  ‘database’ => ‘zarkoDB’);

// $mysqlParams[‘user’] is ‘zarko’
// $mysqlParams[‘password’] is ‘secret’
// $mysqlParams[0] is ???

MySQL Connection in PHP

$myCon = mysql_connect($host, $username, $password)

• Standard practice is to define the values of the
  params in another file.
• Also standard is error handling:
  or die(‘Could not connect: ‘.mysql_error());

Selecting Database

mysql_select_db($database,$myCon)
or die(‘Could not select database’);

• If you omit $myCon, the last open connection
  will be used.
Queries in PHP

```php
$query = 'SELECT bar, beer, price FROM Sells';
$result = mysql_query($query)
or die("Query $query failed: ".mysql_error());
```

- `mysql_query` takes a string argument and returns a result or generates an error.

Cursors in PHP

- `mysql_fetch_row`
- `mysql_fetch_assoc`
- `mysql_fetch_array`

- When applied to the result of a query they return the next tuple as a numeric array, associative array, or both.
- Return false after the last tuple.

Cursor Example

```php
while ($tuple = mysql_fetch_array($result)) {
    echo $tuple[0];
    echo $tuple['beer'];
}
```

Little Bobby Tables

![Little Bobby Tables Comic](http://xkcd.com/327/)

SQL Injection

- Hack input as to change the query that gets executed.
- Usually includes `'` followed by the hacker query followed by `--` to treat anything after it as comments.
- User input should not be trusted!

Bobby Table Revealed

```php
$newUserQ = "INSERT INTO Students VALUES ('$newUserName')";
```

- For Bobby, it evaluates to:
  `INSERT INTO Users VALUES ('Robert'); DROP TABLE Students ;--')";`
- It won’t work with `mysql_query`!