Topics of Discussion

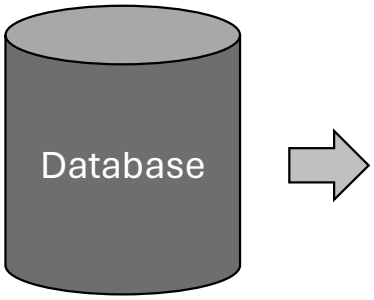
- Databases and their role in the Web
- MySQL Database
- Connecting to a database
- Handling connection errors

Databases and the Web

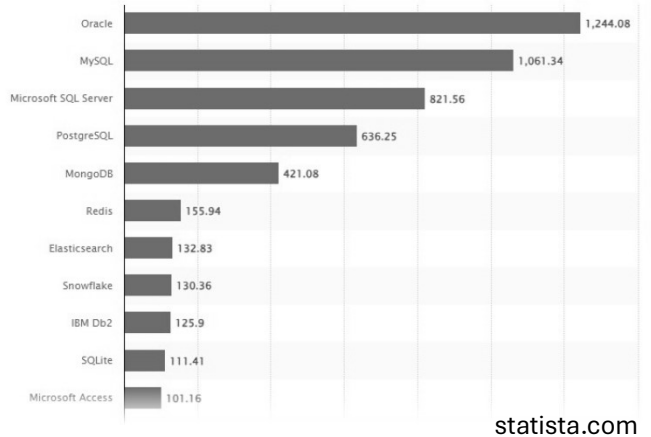
- Dynamic websites make use of some type of back-end data source
  - The most common is a database
- Large websites (high traffic) have a dedicated database server
  - A computer that runs a relational DBMS
- Small websites (low traffic) can get away with the DBMS on the same computer as the web server (as we do in the labs)

Databases and the Web

- Databases play an essential part as they allow for dynamic content
- Data can be stored in the database and then added to the web page as needed



Relational DBMS



statista.com

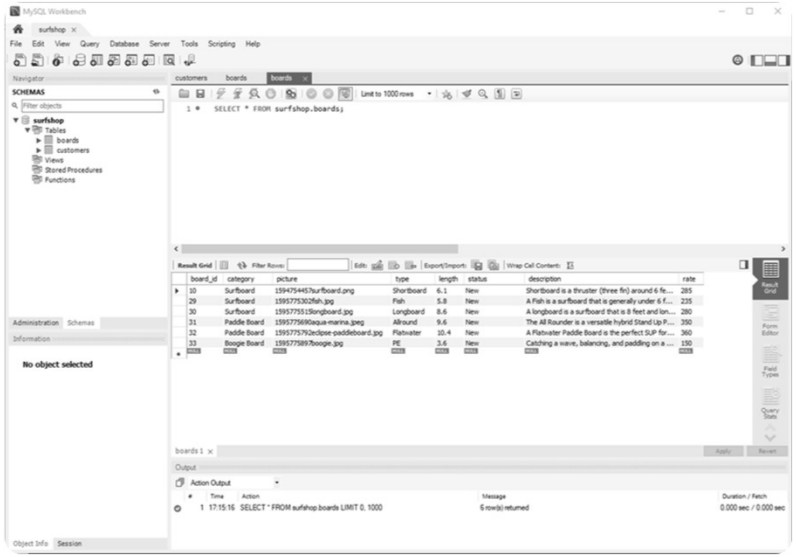
- Many different solutions available
- MySQL works well with PHP
- Use SQL to communicate with DBMS
- MySQL is an open-source relational database management system (DBMS)
  - One of the most popular DBMS
  - Use by companies such as Facebook, Netflix, Uber, Shopify, and mor

MySQL™

W3Schools: MySQL Database

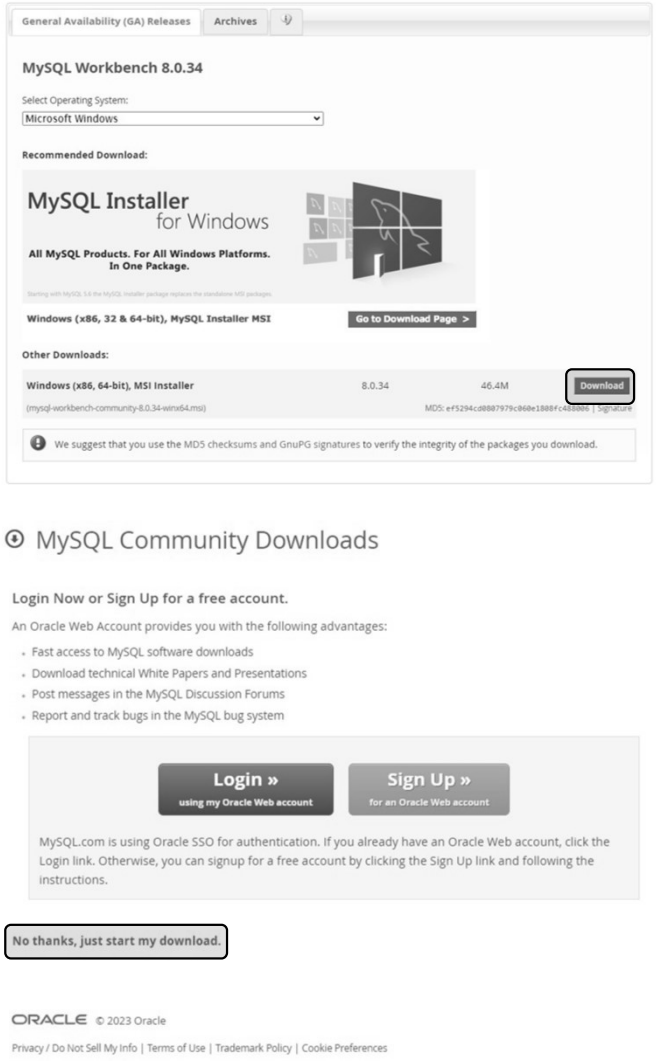
Managing a MySQL Database

- MySQL Workbench
  - Free tool from Oracle
  - Visual interface
  - Already installed in the Hamilton Labs

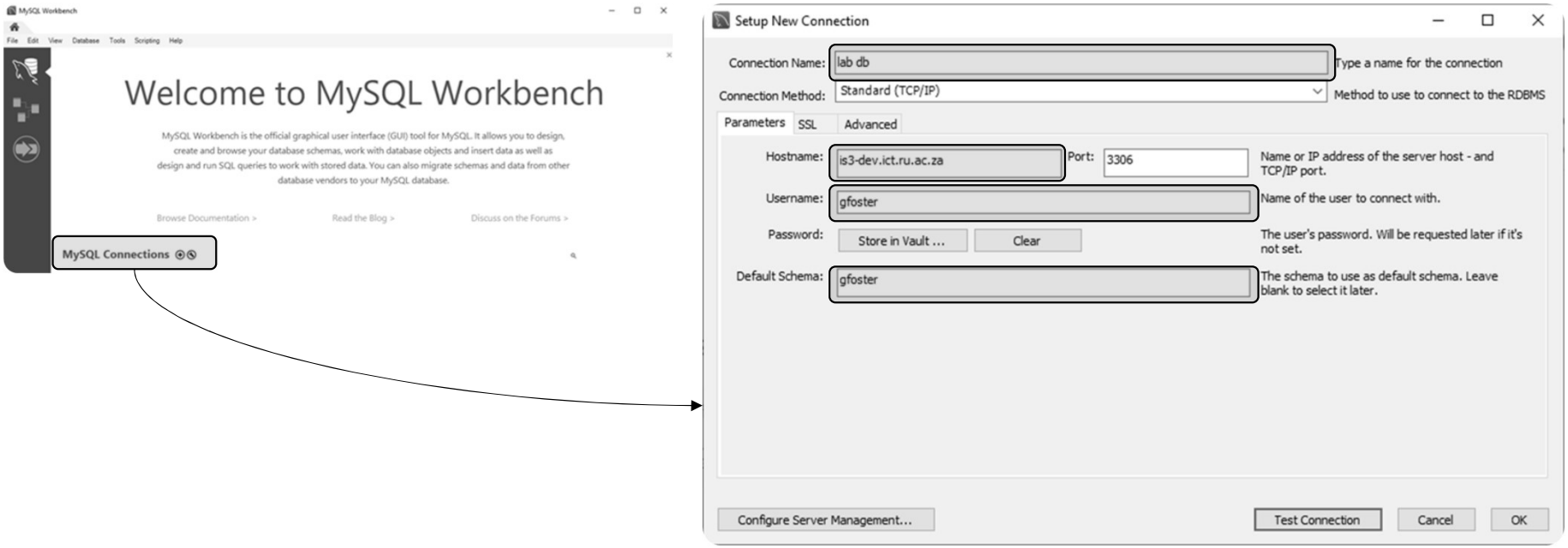


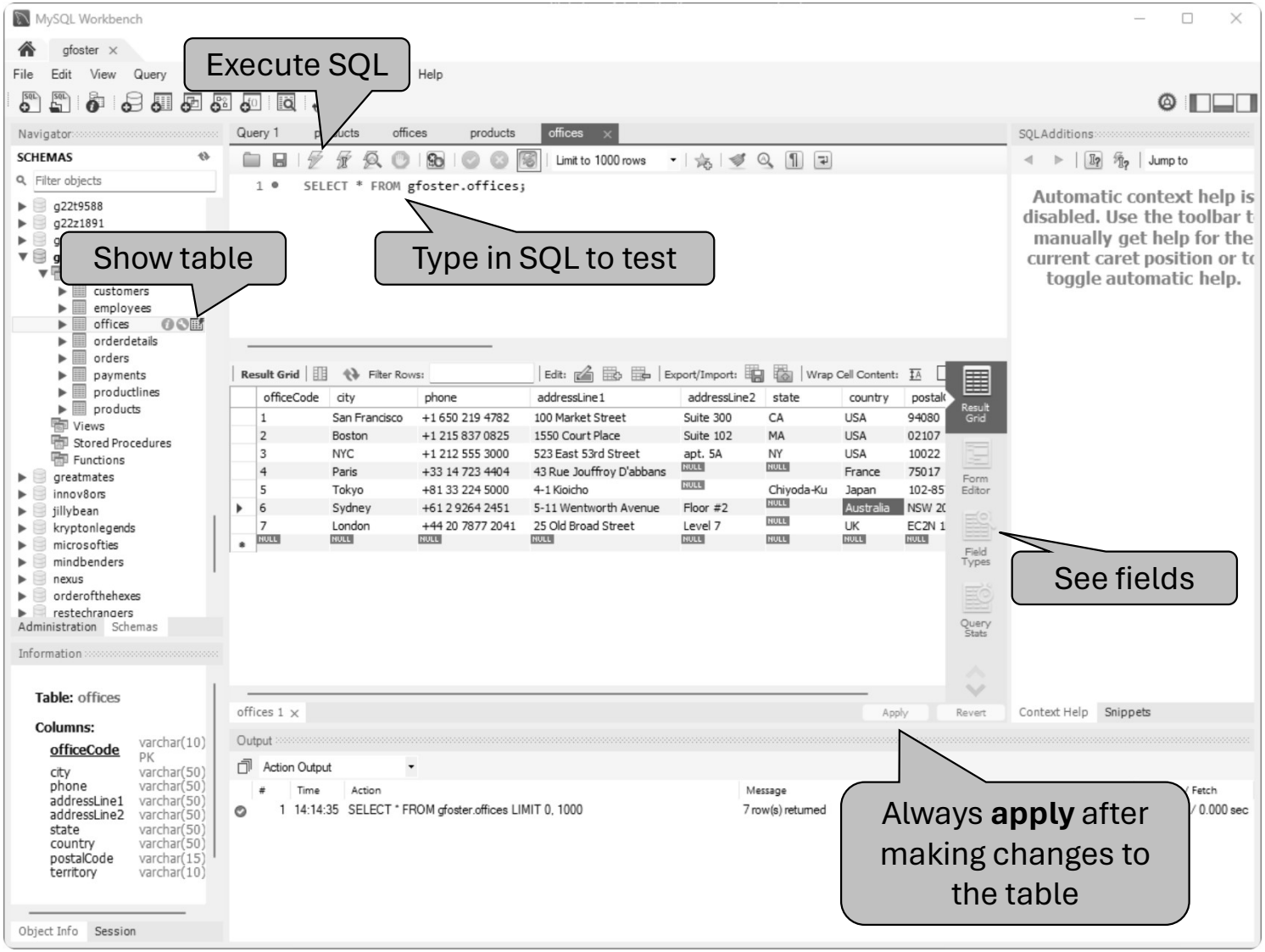
Managing a MySQL Database

- No need to Login or Sign Up
  - https://dev.mysql.com/downloads/workbench/
- May also need to install Visual C++ Redistributable
  - https://aka.ms/vs/17/release/vc_redist.x64.exe



Connecting to the Lab Database




MySQL Workbench
gfoster
File Edit View Query Help
Execute SQL
Navigator
SCHEMAS
Filter objects
g22t9588
g22z1891
customers
employees
offices
orderdetails
orders
payments
productlines
products
Views
Stored Procedures
Functions
greatmates
innov8ors
jillybean
kryptonlegends
microsofties
mindbenders
nexus
orderofthehexes
restechrangers
Administration Schemas
Information
Table: offices
Columns:
officeCode varchar(10) PK
city varchar(50)
phone varchar(50)
addressLine1 varchar(50)
addressLine2 varchar(50)
state varchar(50)
country varchar(50)
postalCode varchar(15)
territory varchar(10)
Object Info Session
Query 1
offices
products
offices
Limit to 1000 rows
1 SELECT * FROM gfoster.offices;
Show table
Type in SQL to test
Result Grid
Filter Rows:
Edit:
Export/Import:
Wrap Cell Content:
officeCode city phone addressLine1 addressLine2 state country postal
1 San Francisco +1 650 219 4782 100 Market Street Suite 300 CA USA 94080
2 Boston +1 215 837 0825 1550 Court Place Suite 102 MA USA 02107
3 NYC +1 212 555 3000 523 East 53rd Street apt. 5A NY USA 10022
4 Paris +33 14 723 4404 43 Rue Jouffroy D'abbans NULL NULL France 75017
5 Tokyo +81 33 224 5000 4-1 Kioicho NULL Chiyoda-Ku Japan 102-85
6 Sydney +61 2 9264 2451 5-11 Wentworth Avenue Floor #2 NULL Australia NSW 20
7 London +44 20 7877 2041 25 Old Broad Street Level 7 NULL UK EC2N 1
Result Grid
Form Editor
Field Types
Query Stats
See fields
offices 1
Apply
Revert
SQLAdditions
Jump to
Automatic context help is disabled. Use the toolbar t manually get help for the current caret position or to toggle automatic help.
Context Help Snippets
Output
Action Output
Time Action Message Fetch
1 14:14:35 SELECT * FROM gfoster.offices LIMIT 0, 1000 7 row(s) returned / 0.000 sec
Always apply after making changes to the table

PHP and Databases

- PHP scripts can handle SQL commands to insert, retrieve or modify data from a database
- The MySQLi API allows you to access the functionality provided by MySQL
- First you must have a connection to the database
- Once a connection has been established, SQL commands can be issued to insert, retrieve or modify data

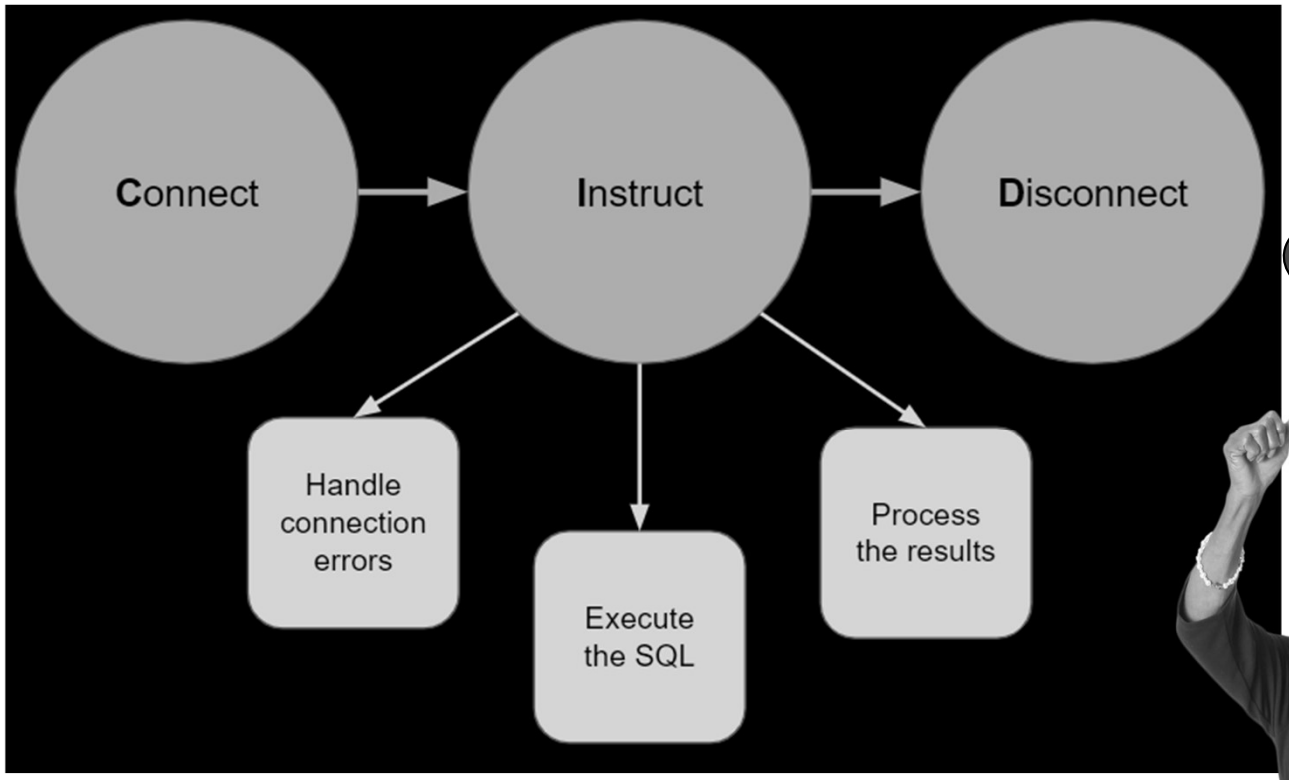
mysqli Class

- Make use of the `mysqli` class, which represents a connection between PHP and a MySQL database

mysqli	
`$connect_error`	• Returns a description of the last connection error.
`close()`	• Closes a previously opened database connection.
`prepare()`	• Prepares an SQL statement for execution.
`query()`	• Performs a query on the database.
`real_escape_string()`	• Escapes special characters in a string for use in an SQL statement.

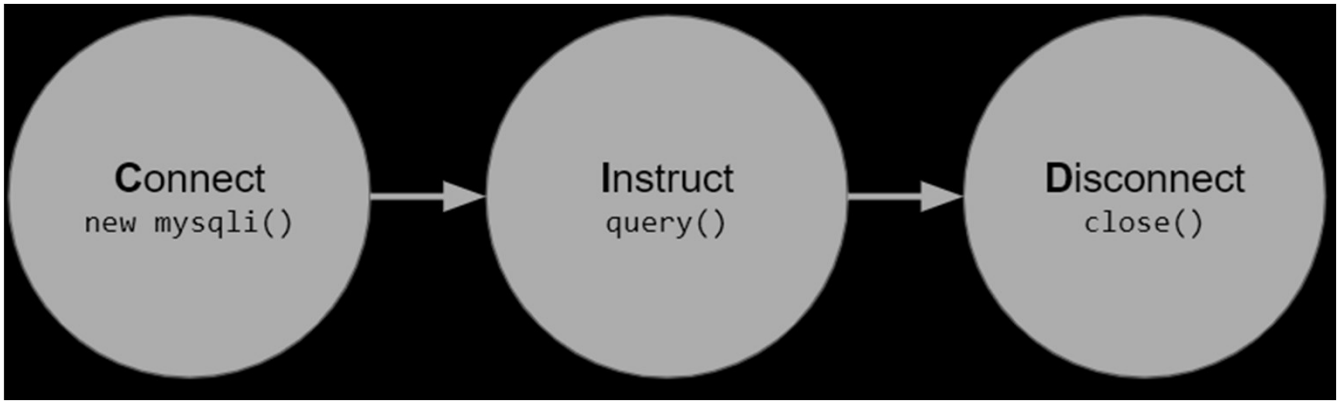
PHP Manual: mysqli class

PHP and Databases



PHP Database Functions

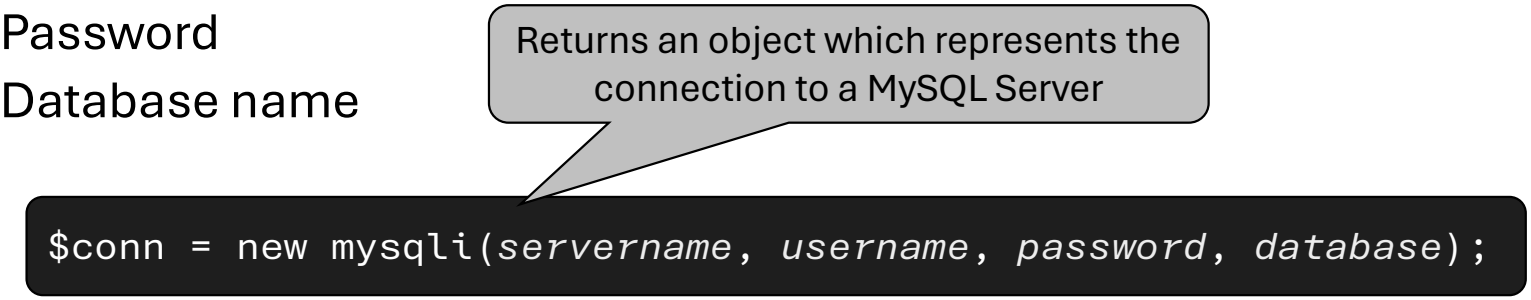
- Two main methods:
  - `query()`
  - `close()`



W3Schools: PHP MySQLi

mysqli Object

- Need to instantiate a mysqli object to open a connection to the MySQL Server
- Use the new keyword to instantiate objects
- Must provide four pieces of information:
  - Database server name
  - Username
  - Password
  - Database name



W3Schools: MySQL Connect

query() Method

- Performs a query against the database
- Must provide the SQL commands (query string)
  - SQL code (e.g., SELECT * FROM customers)

Returns **false** on failure.

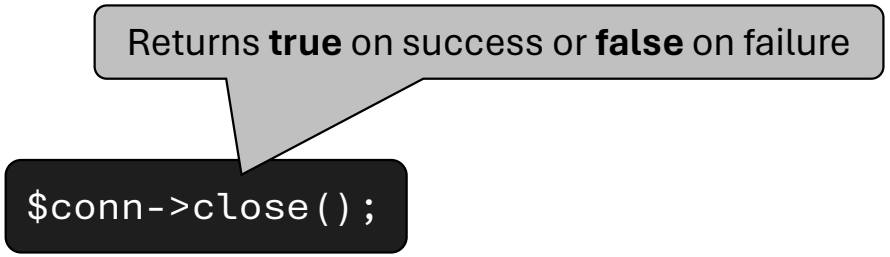
For successful queries which produce a result set, such as SELECT, query() will return a **mysqli_result** object.

For other successful queries, query() will return **true**

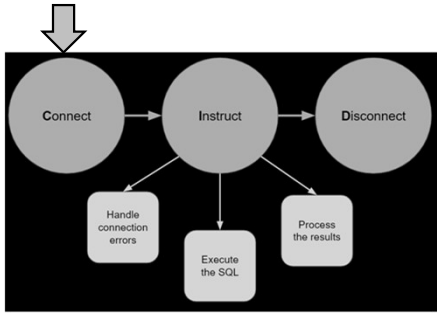
```
$result = $conn->query(sql);
```

close() Method

- Closes a previously opened database connection
- Default number of simultaneous connections for MySQL = 151
- Need to conserve them, hence it is good practice to close connection as soon as it's no longer needed

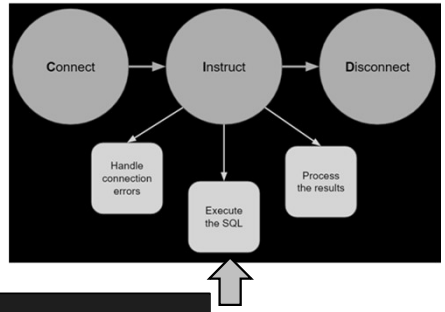


Make the Connection



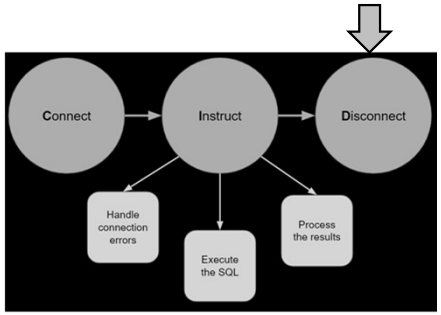

```php
<?php
// store database credentials
$servername = "is3-dev.ict.ru.ac.za";
$username = "gfoster";
$password = "is302";
$database = "gfoster";
// make connection to database
$conn = new mysqli($servername, $username, $password, $database);
// issue query instructions
$sql = "SELECT * FROM products";
$result = $conn->query($sql);
// close the connection to database
$conn->close();
?>
```

Execute Instructions



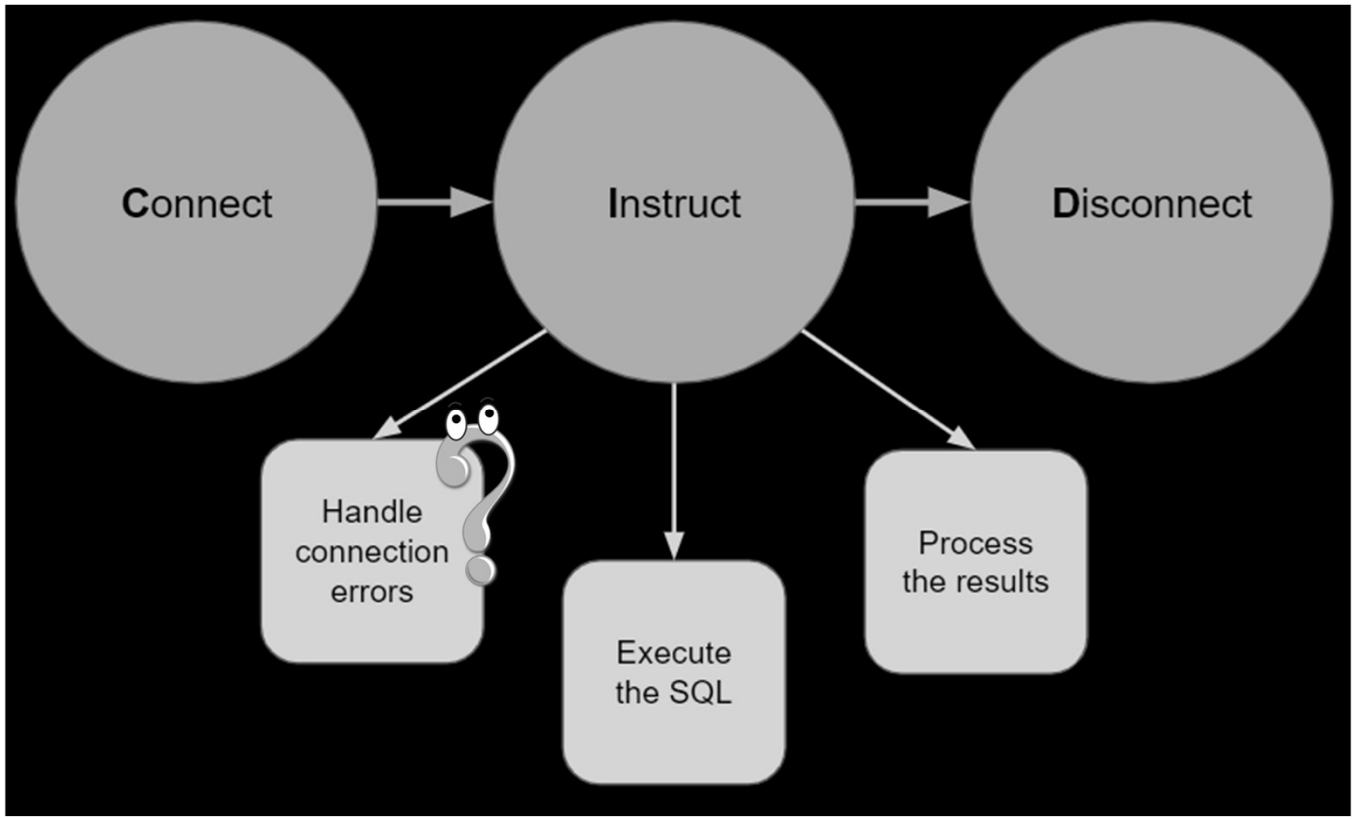

```php
<?php
// store database credentials
$servername = "is3-dev.ict.ru.ac.za";
$username = "gfoster";
$password = "is302";
$database = "gfoster";
// make connection to database
$conn = new mysqli($servername, $username, $password, $database);
// issue query instructions
$sql = "SELECT * FROM products";
$result = $conn->query($sql);
// close the connection to database
$conn->close();
?>
```

Close the Connection




```php
<?php
// store database credentials
$servername = "is3-dev.ict.ru.ac.za";
$username = "gfoster";
$password = "is302";
$database = "gfoster";
// make connection to database
$conn = new mysqli($servername, $username, $password, $database);
// issue query instructions
$sql = "SELECT * FROM products";
$result = $conn->query($sql);
// close the connection to database
$conn->close();
?>
```

Error Handling



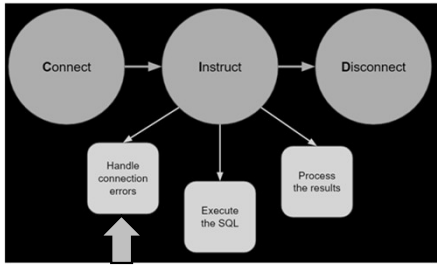
Error Handling

- Useful when something is wrong with the connection arguments, or the server cannot be reached
- Provide information about the error
- Need to cater for two situations:
  - Cannot connect to the database
  - Something goes wrong when executing the query instructions

Error Handling

- Cannot connect to the database
  - Need to make use of the `connect_error` property
  - Returns the error message from the last connection attempt
    - null is returned if no error occurred
  - Terminate further execution of script with PHP's built-in `die()` function
- Something goes wrong when executing the query instructions
  - Need to test if query executed (returns FALSE if it did not)
  - Terminate further execution of script with PHP's built-in `die()` function

Error Handling

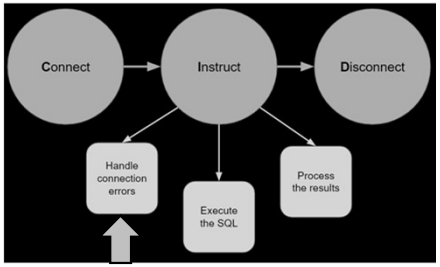


- Cannot connect to the database

Connection failed: Access denied for user 'gfoster'@'is3-dev.ict.ru.ac.za' (using password: YES)

```php
<?php
// store database credentials
$servername = "is3-dev.ict.ru.ac.za";
$username = "gfoster";
$password = "is302x";
$database = "gfoster";
// make connection to database
$conn = new mysqli($servername, $username, $password, $database);
// check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
// issue query instructions
$sql = "SELECT * FROM products";
$result = $conn->query($sql);
// close the connection to database
$conn->close();
?>
```

Error Handling



- Something goes wrong when executing the query instructions

x is equal to y, and they are of the **same type**

```php
<?php
// store database credentials
$servername = "is3-dev.ict.ru.ac.za";
$username = "gfoster";
$password = "is302";
$database = "gfoster";
// make connection to database
$conn = new mysqli($servername, $username, $password, $database);
// check connection successful
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
// issue query instructions
$sql = "SELECT * FROM productsX";
$result = $conn->query($sql);
//check query successful
if ($result === FALSE) {
    die("Query failed to execute");
}
// close the connection to database
$conn->close();
?>
```

Query failed to execute

Error Handling

- You can add HTML/CSS to the string you are giving to the `die()` function to format it

Connection failed: Access denied for user 'gfoster'@'is3-dev.ict.ru.ac.za' (using password: YES)

```php
<?php
// store database credentials
$servername = "is3-dev.ict.ru.ac.za";
$username = "gfoster";
$password = "is302x";
$database = "gfoster";
// make connection to database
$conn = new mysqli($servername, $username, $password, $database);
// check connection successful
if ($conn->connect_error) {
    die("<p class=\"error\"><strong>Connection failed:</strong> " . $conn->connect_error . "</p>");
}
// issue query instructions
```

```css
<style>
.error {
    color: red;
}
</style>
```

Storing Connection Details

- Hard-coding the database connection details in your code is not ideal
- Connection details may change as a site moves or gets upgraded
- Place details in a central location so we only need to change one file - not every web page
- Insert one PHP script into another PHP script
  - Reuse code as needed in other scripts
    - write once and use many places

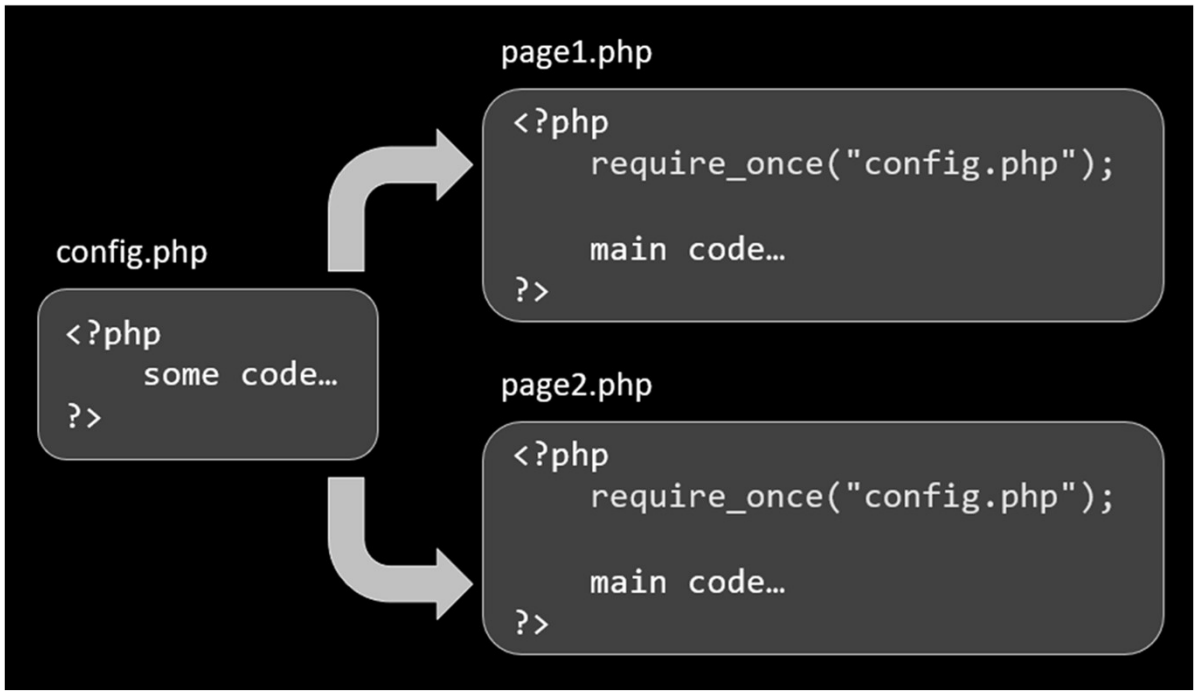
Storing Connection Details

- Define constants that store the database connection details that are in a separate file
  - No $ sign before the constant name
  - Convention is to use ALL CAPITALS for constant names
- Automatically global across the entire script
- Include files that allow you to share code in other files
- Use the PHP built-in `require_once()` function to include the file as needed

W3Schools: PHP Constants

Storing Connection Details

- Using include files



Storing Connection Details

```php
<?php
// add the constants from file
require_once("config.php");
// make connection to database
$conn = new mysqli(SERVERNAME, USERNAME, PASSWORD, DATABASE);
// check connection successful
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
// issue query instructions
$sql = "SELECT * FROM products";
$result = $conn->query($sql);
//check query successful
if ($result === FALSE) {
    die("Query failed to execute");
}
// close the connection to database
$conn->close();
?>
```

config.php

```php
<?php
define("SERVERNAME", "is3-dev.ict.ru.ac.za");
define("USERNAME", "gfoster");
define("PASSWORD", "is302");
define("DATABASE", "gfoster");
```

Note: using constants now