Databases, MySQL & PHP

Managing data

Building Data Dynamic Web Sites

- Truly dynamic web sites
  - Content changes over time
  - Content customised for individual user
  - Content automatically generated
- Content Programmatically generated
  - Can be File system based
    - HTML and Images stored on File System
      - Gets hard to manage over time
  - Database based
    - HTML, Images etc all generated from database
      - Easier to manage
      - If data is too large, can overload the database

Database?

Database

- **Structured collection of data.**
  - **Tables**
  - **Fields**
  - **Query**
  - **Reports**
- **Essentially a much more sophisticated implementation of the flat files.**

Relational Database

Relational Database

- Stores data in separate tables instead of a single store.
- Relationships between tables are set
- In theory, this provides a faster, more flexible database system.

Example

• We wish to maintain a **database** of student names, IDs, addresses, and any other information.

• Will be **updated frequently** with new names and information.

• Will want to **retrieve data** based on some predicate.

- **e.g, 'give me the names of all Massey students who live in Albany'.**

• Will want to update database with new information about students, not previously recorded.

- **e.g., may decide we want to include IRD nos.**

• Very difficult to manage using 'flat file' systems

Databases

- **Fast, Efficient back end storage**
  - Easier to manage than file system based approach
- **Relational Database structure**
  - Well developed theory and practise
- **Multi-user capable**
  - Multithreaded, multiprocessor, sometimes cluster based systems
- **Standards based queries**
  - **Structured Query Language (SQL)**

MySQL Database

- world's most popular open source database because of its consistent **fast performance, high reliability and ease of use**
- Open Source License:- free
  - GNU General Public License
  - Free to modify and distribute but all modification must be available in source code format
- Commercial:- not free
  - Fully paid up professional support
- **used by Google, Facebook Nokia, YouTube, Yahoo!, Alcatel-Lucent, Zappos.com, etc.**

Basic Database Server Concepts

- **Database runs as a server**
  - Attaches to either a default port or an administrator specified port
- **Clients connect to database**
  - For secure systems
    - authenticated connections
    - usernames and passwords
- **Clients make queries on the database**
  - Retrieve content
  - Insert content
- **SQL (Structured Query Language)** is the language used to insert and retrieve content

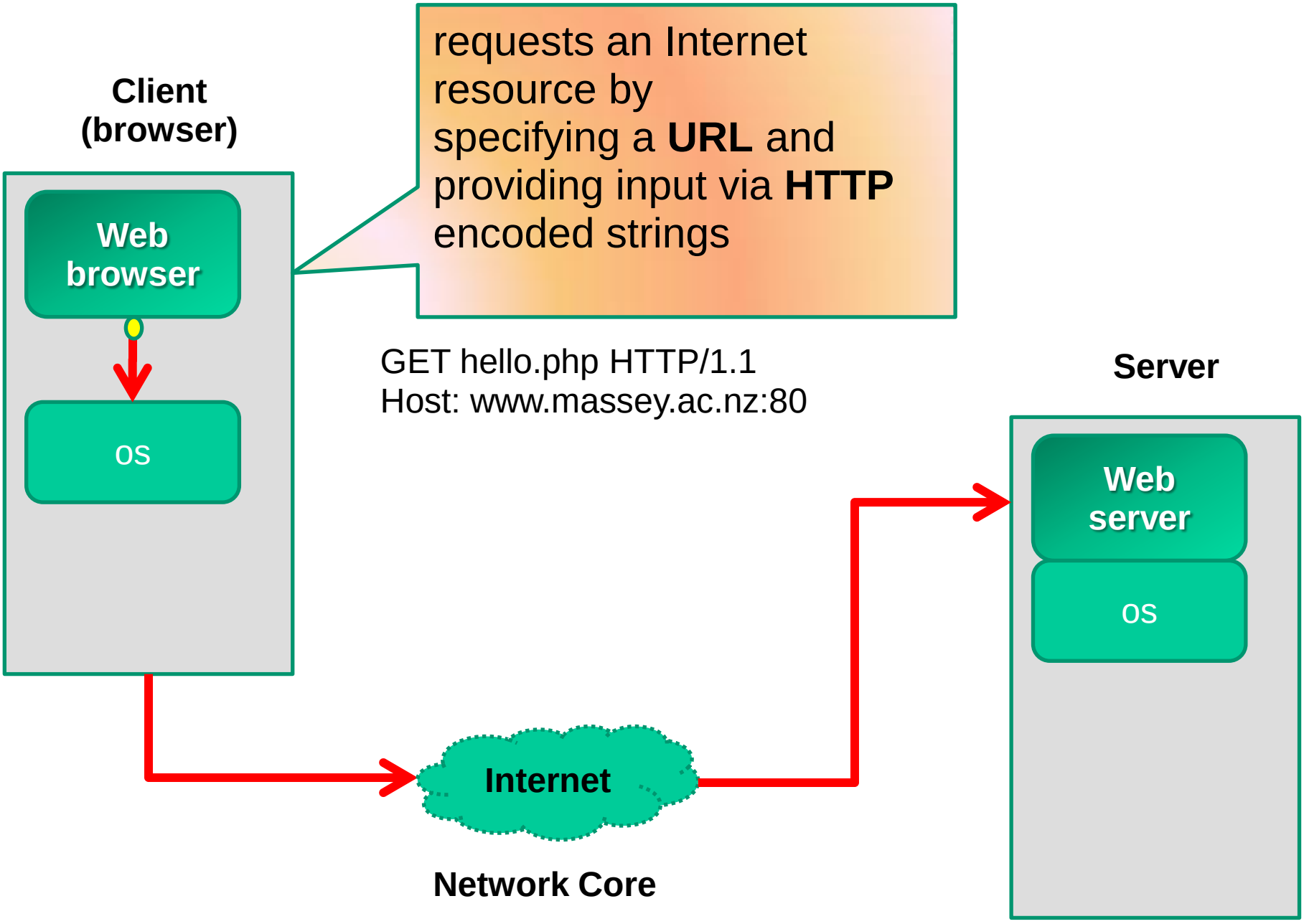
Database Management System?

Database Management System

- **Manages the storage and retrieval of data to and from the database and hides the complexity of what is actually going on from the user.**

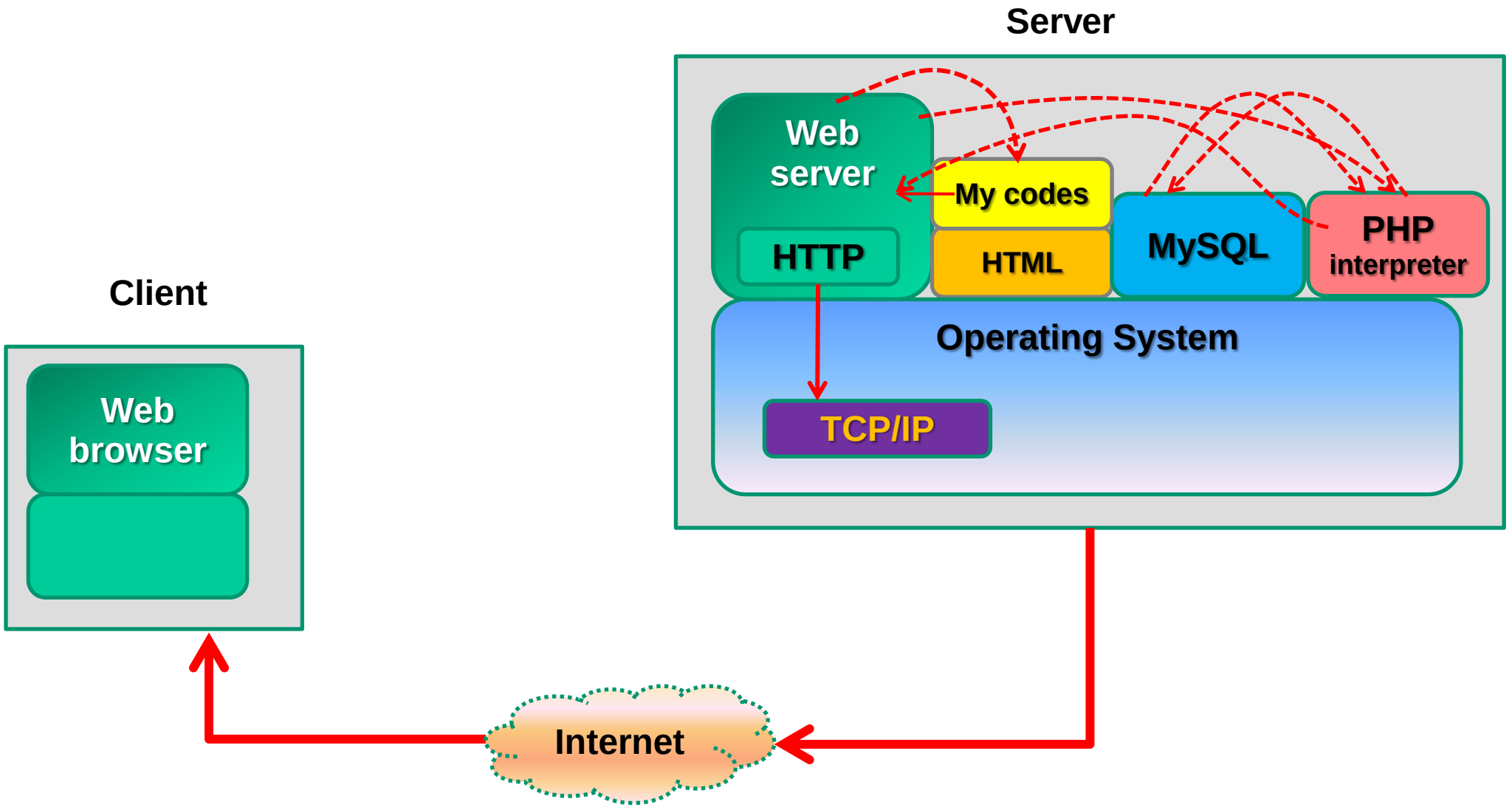
- **MySQL is a relational database management system**

Client: makes a request

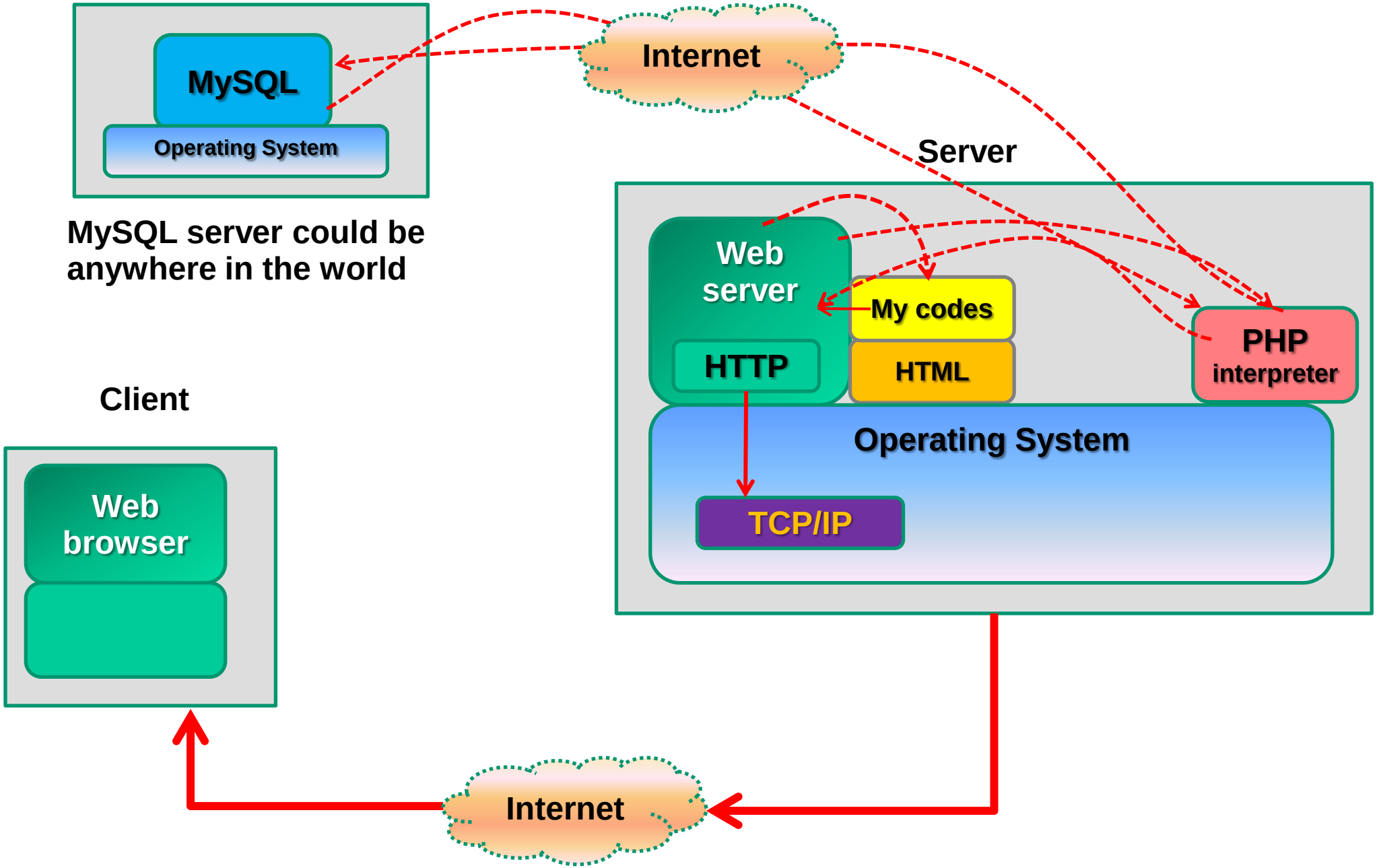


Server: responds

- Webserver supports HTTP.



Server: responds



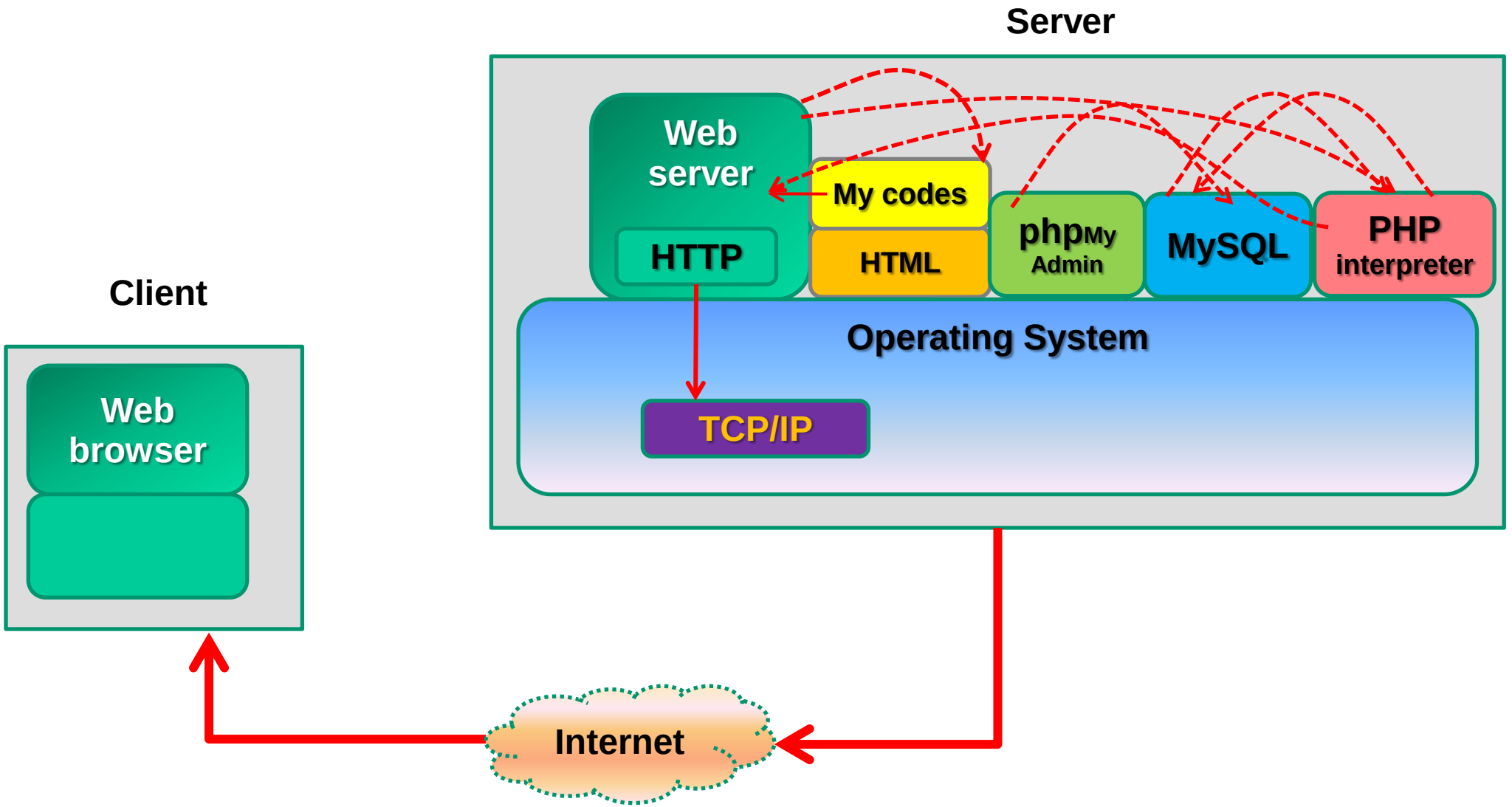
phpMyAdmin

- **MySQL can be controlled through a simple command-line interface; however, we can use phpMyAdmin as an interface to MySQL.**

- **phpMyAdmin is a very powerful tool; it provides a large number of facilities for customising a database management system.**

Server: responds

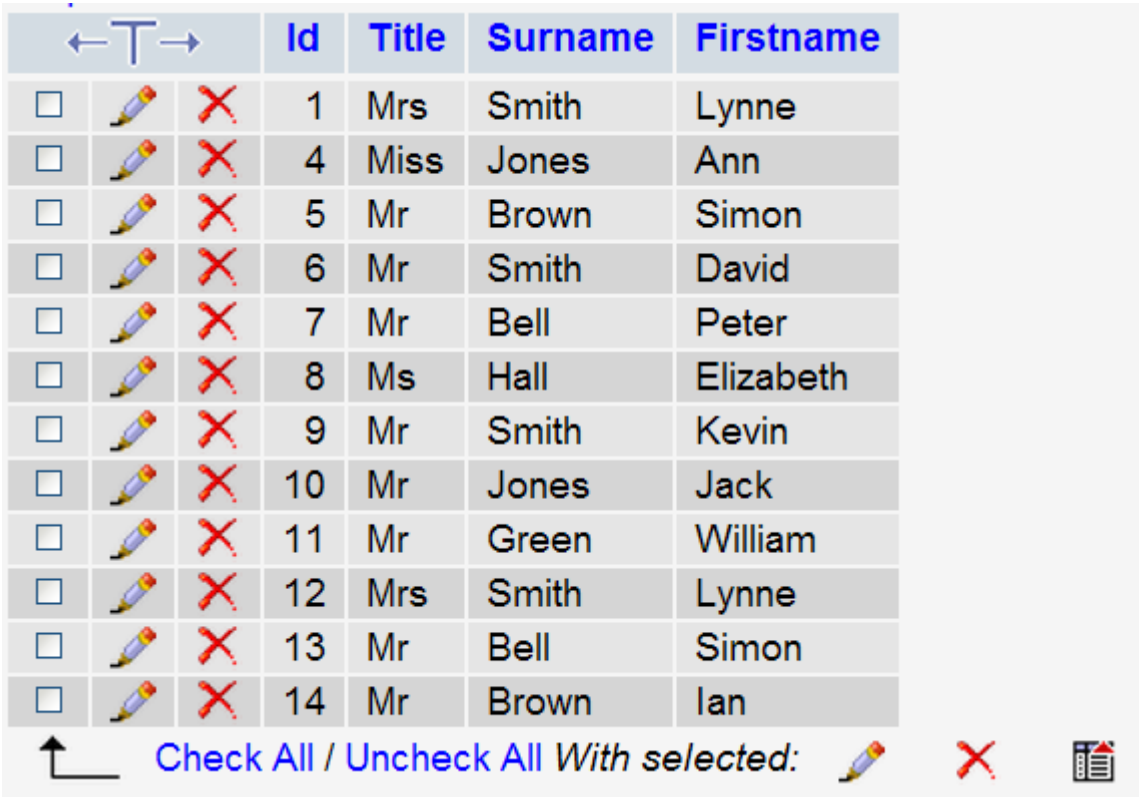
- Webserver supports HTTP.



Database Example

- **A Quick Tour**

Table: Customers (data)



←T→	Id	Title	Surname	Firstname
☐ ✎ ✕	1	Mrs	Smith	Lynne
☐ ✎ ✕	4	Miss	Jones	Ann
☐ ✎ ✕	5	Mr	Brown	Simon
☐ ✎ ✕	6	Mr	Smith	David
☐ ✎ ✕	7	Mr	Bell	Peter
☐ ✎ ✕	8	Ms	Hall	Elizabeth
☐ ✎ ✕	9	Mr	Smith	Kevin
☐ ✎ ✕	10	Mr	Jones	Jack
☐ ✎ ✕	11	Mr	Green	William
☐ ✎ ✕	12	Mrs	Smith	Lynne
☐ ✎ ✕	13	Mr	Bell	Simon
☐ ✎ ✕	14	Mr	Brown	Ian

Check All / Uncheck All *With selected:* ✎ ✕

Table: Products (data)

	Id	Name	Description	Quantity	Cost
☐	1	Beer Glass	600 ml Beer Glass	345	3.99
☐	2	Wine Glass	125 ml Wine Glass	236	2.99
☐	3	Wine Glass	175 ml Wine Glass	436	3.5
☐	4	Shot Glass	50 ml Small Glass	132	1.5
☐	5	Spirit Glass	100 ml Short Glass	489	2.5
☐	6	Long Glass	200 ml Tall Glass	263	2.5
☐	7	Beer Glass	300 ml Beer Glass	247	2.99
☐	8	Wine Glass	225 ml Wine Glass	96	3.99

Check All / Uncheck All *With selected:*

Table: Purchases (data)

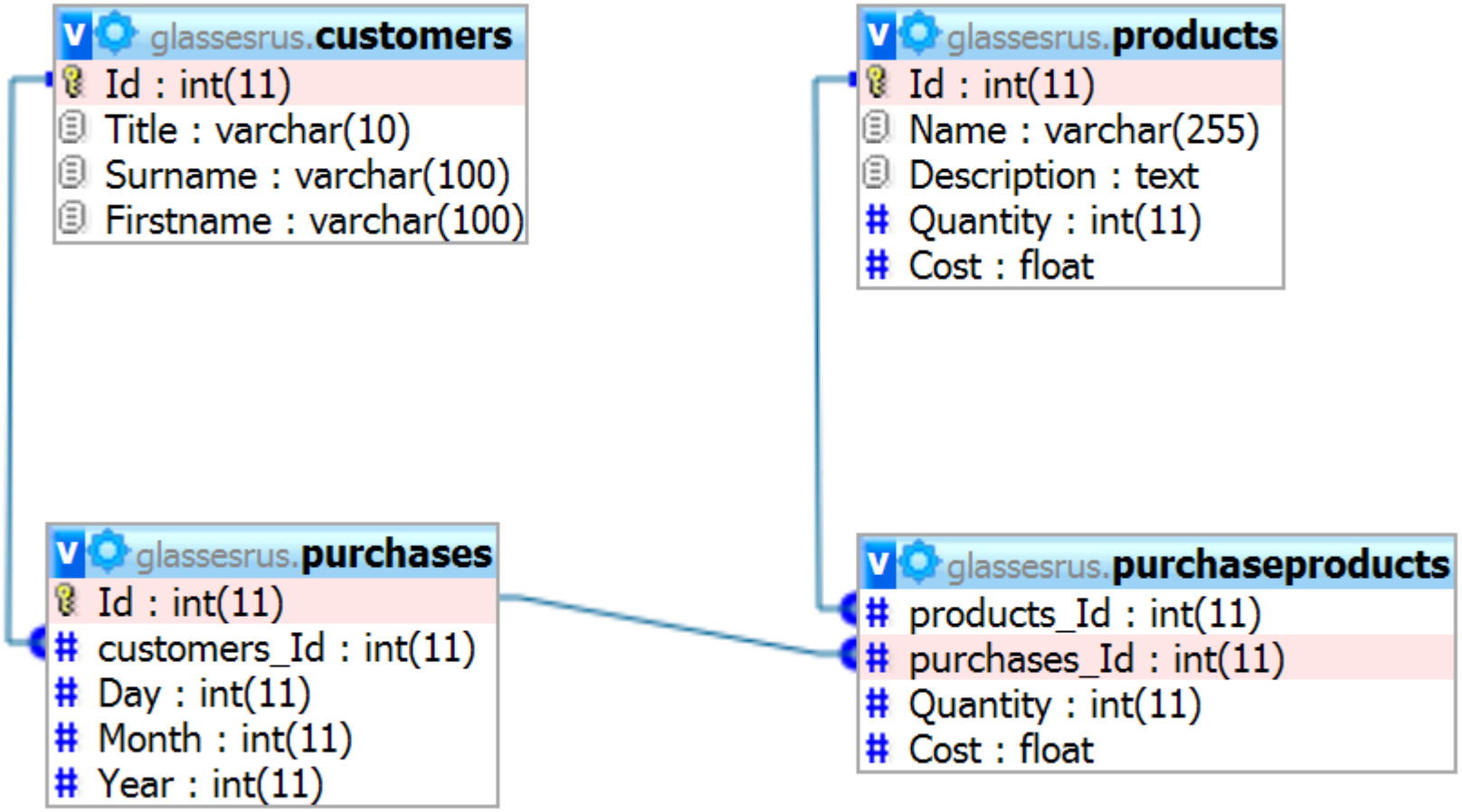
←T→	Id	customers_Id	Day	Month	Year
☐ ✎ ✗	1	2	3	9	2005
☐ ✎ ✗	2	4	6	9	2005
☐ ✎ ✗	3	6	13	9	2005
☐ ✎ ✗	4	2	22	9	2005
☐ ✎ ✗	5	1	28	9	2005
☐ ✎ ✗	6	9	1	10	2005
☐ ✎ ✗	7	7	1	10	2005

Check All / Uncheck All *With selected:* ✎ ✗

Table: PurchaseProducts (data)

←T→	products_Id	purchases_Id	Quantity	Cost
☐ ✎ ✕	2	1	20	2.99
☐ ✎ ✕	3	2	10	3
☐ ✎ ✕	8	2	30	4.5
☐ ✎ ✕	6	3	25	2.5
☐ ✎ ✕	3	4	10	3.5
☐ ✎ ✕	4	4	100	1.5
☐ ✎ ✕	5	4	40	3
☐ ✎ ✕	1	5	22	3.99
☐ ✎ ✕	1	6	5	3.99
☐ ✎ ✕	3	7	15	3.5
☐ ✎ ✕	4	7	25	2
☐ ✎ ✕	5	7	10	2.5
☐ ✎ ✕	7	7	55	2.5
☐ ✎ ✕	8	7	1	3.99
↑ Check All / Uncheck All *With selected:*			✎ ✕	

Database Design



Database Field Types

In MySQL there are three main types :

- **text**
- **number**
- **Date/Time.**

Text Field Types

CHAR(size)	Holds a fixed length string (can contain letters, numbers, and special characters). The fixed size is specified in parenthesis. Can store up to **255** characters
VARCHAR(size)	Holds a variable length string (can contain letters, numbers, and special characters). The maximum size is specified in parenthesis. Can store up to **255** characters. **Note:** If you put a greater value than 255 it will be converted to a TEXT type
TINYTEXT	Holds a string with a maximum length of **255** characters
TEXT	Holds a string with a maximum length of **65,535** characters
MEDIUMTEXT	Holds a string with a maximum length of **16,777,215** characters
LONGTEXT	Holds a string with a maximum length of **4,294,967,295** characters
ENUM(x,y,z,etc.)	Let you enter a list of possible values. You can list up to 65535 values in an ENUM list. If a value is inserted that is not in the list, a blank value will be inserted.**Note:** The values are sorted in the order you enter them. You enter the possible values in this format: **ENUM('X','Y','Z')**

http://www.w3schools.com/sql/sql_datatypes.asp

Numeric Field Types

TINYINT(size)	**-128 to 127** normal. 0 to 255 UNSIGNED*. The maximum number of digits may be specified in parenthesis
SMALLINT(size)	**-32768 to 32767** normal. 0 to 65535 UNSIGNED*. The maximum number of digits may be specified in parenthesis
MEDIUMINT(size)	**-8388608 to 8388607** normal. 0 to 16777215 UNSIGNED*. The maximum number of digits may be specified in parenthesis
INT(size)	**-2147483648 to 2147483647** normal. 0 to 4294967295 UNSIGNED*. The maximum number of digits may be specified in parenthesis
BIGINT(size)	**-9223372036854775808 to 9223372036854775807** normal. 0 to 18446744073709551615 UNSIGNED*. The maximum number of digits may be specified in parenthesis
FLOAT(size,d)	A small number with a floating decimal point. The maximum number of digits may be specified in the size parameter. The maximum number of digits to the right of the decimal point is specified in the d parameter
DOUBLE(size,d)	A large number with a floating decimal point. The maximum number of digits may be specified in the size parameter. The maximum number of digits to the right of the decimal point is specified in the d parameter
DECIMAL(size,d)	A DOUBLE **stored as a string** , allowing for a fixed decimal point. The maximum number of digits may be specified in the size parameter. The maximum number of digits to the right of the decimal point is specified in the d parameter

http://www.w3schools.com/sql/sql_datatypes.asp

Date and Time Field Types

DATE()	A date. Format: YYYY-MM-DD**Note:** The supported range is from '1000-01-01' to '9999-12-31'
DATETIME()	*A date and time combination. Format: YYYY-MM-DD HH:MM:SS**Note:** The supported range is from '1000-01-01 00:00:00' to '9999-12-31 23:59:59'
TIMESTAMP()	*A timestamp. TIMESTAMP values are stored as the number of seconds since the Unix epoch ('1970-01-01 00:00:00' UTC). Format: YYYY-MM-DD HH:MM:SS**Note:** The supported range is from '1970-01-01 00:00:01' UTC to '2038-01-09 03:14:07' UTC
TIME()	A time. Format: HH:MM:SS**Note:** The supported range is from '-838:59:59' to '838:59:59'
YEAR()	A year in two-digit or four-digit format.**Note:** Values allowed in four-digit format: 1901 to 2155. Values allowed in two-digit format: 70 to 69, representing years from 1970 to 2069

http://www.w3schools.com/sql/sql_datatypes.asp

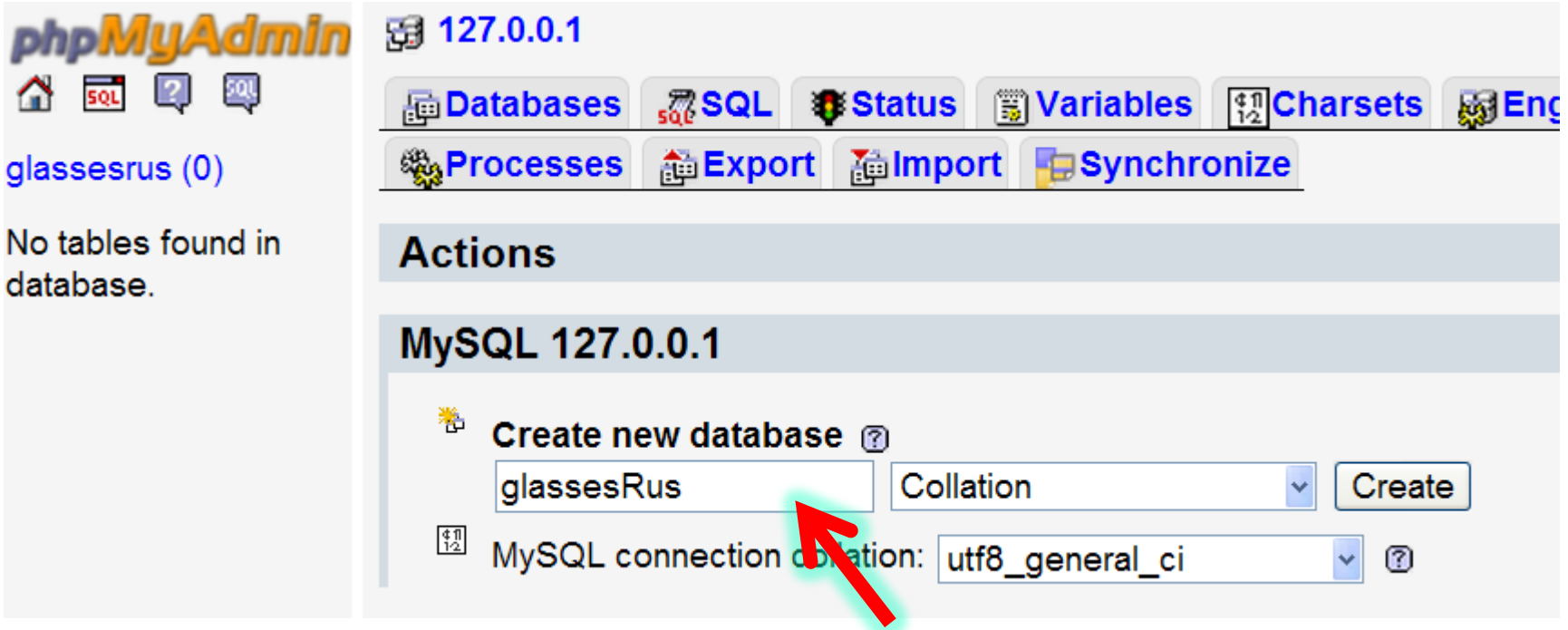
phpMyAdmin

- **A Quick Tour**

phpMyAdmin


EASYPHP
PHP : MYSQL : APACHE FOR WINDOWS
help
english
5.3.3
www.easyphp.org
Support, download, faq, news, forum...
Support this project
PHP 5.3 migration guide
Most improvements in PHP 5.3.x have no impact on existing code.
However, there are a few incompatibilities and new features that should be considered.
If you want to use EasyPHP on an USB key, you just nee
copy the entire EasyPHP folder on the key. Be sure that al
scripts are in the folder 'www' and your databases in 'mysql
PHP 5.3.3
APACHE 2.2.16
MYSQL 5.1.49
PHPMYADMIN 3.3.5
Manage MySQL
MySQL Parameters
PHP Parameters
Time Zone
Extensions
LOCAL WEB
C:\Program Files\EasyPHP-5.3.3\www\
Root
07273320
09213244
phptest
protected
protected2
temp

Create Database



Create Table: Customers

Specify the Table's Fields & Attributes: Customers

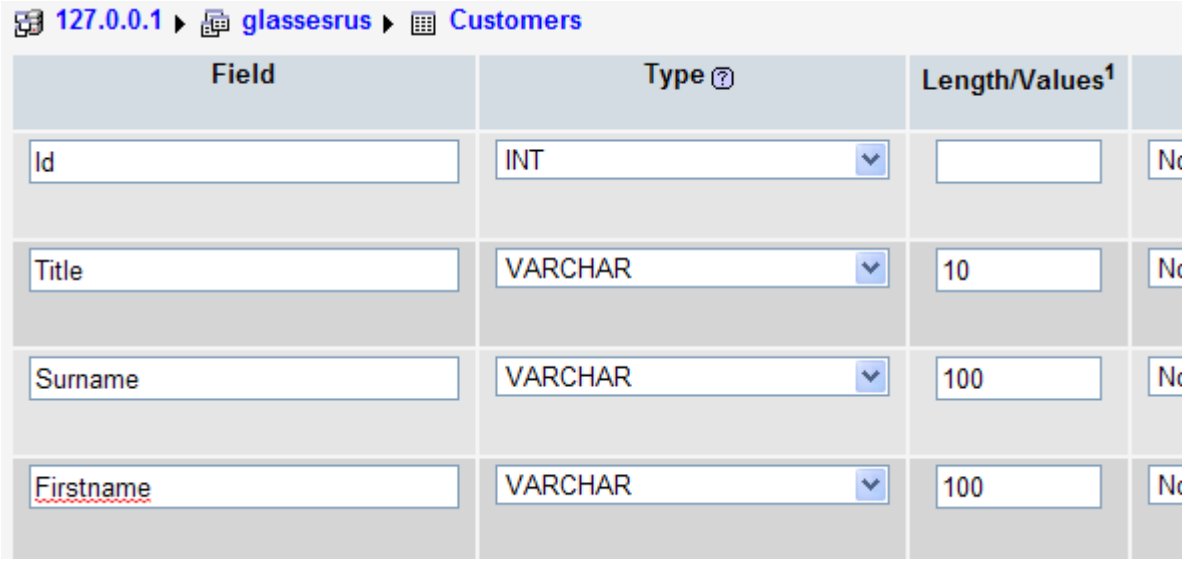


Table Edit Screen: Customers

127.0.0.1 ▸ glassesrus ▸ Customers

Browse | Structure | SQL | Search | Tracking | Insert | Export | Import

Operations | Empty | Drop

Table `glassesrus`.`Customers` has been created.

```
CREATE TABLE `glassesrus`.`Customers` (
  `Id` INT NOT NULL AUTO_INCREMENT ,
  `Title` VARCHAR( 10 ) NOT NULL ,
  `Surname` VARCHAR( 100 ) NOT NULL ,
  `Firstname` VARCHAR( 100 ) NOT NULL ,
  PRIMARY KEY ( `Id` )
) ENGINE = MYISAM ;
```

[Edit] [Create PHP Code]

	Field	Type	Collation	Attributes	Null	Default	Extra		
☐	Id	int(11)			No	*None*	AUTO_INCREMENT		
☐	Title	varchar(10)	latin1_swedish_ci		No	*None*			
☐	Surname	varchar(100)	latin1_swedish_ci		No	*None*			
☐	Firstname	varchar(100)	latin1_swedish_ci		No	*None*			

Check All / Uncheck All *With selected:*

Table: Products

127.0.0.1 ▸ glassesrus ▸ Products

Field	Type	Length/Values[1]
Id	INT	
Name	VARCHAR	255
Description	TEXT	
Quantity	INT	
Cost	FLOAT	

Table: Products

Table `glassesrus`.`Products` has been created.

```
CREATE TABLE `glassesrus`.`Products` (
  `Id` INT NOT NULL AUTO_INCREMENT PRIMARY KEY ,
  `Name` VARCHAR( 255 ) NOT NULL ,
  `Description` TEXT NOT NULL ,
  `Quantity` INT NOT NULL ,
  `Cost` FLOAT NOT NULL
) ENGINE = MYISAM ;
```

[Edit] [Create PHP Code]

	Field	Type	Collation	Attributes	Null	Default	Extr
☐	**Id**	int(11)			No	*None*	AUTO_INCR
☐	**Name**	varchar(255)	latin1_swedish_ci		No	*None*	
☐	**Description**	text	latin1_swedish_ci		No	*None*	
☐	**Quantity**	int(11)			No	*None*	
☐	**Cost**	float			No	*None*	

Insert Record: Customers

```
INSERT INTO `glassesrus`.`customers` (
  `Id` ,
  `Title` ,
  `Surname` ,
  `Firstname`
)
VALUES (
  NULL , 'Mrs', 'Smith', 'Lynne'
);
```

[Edit] [Create

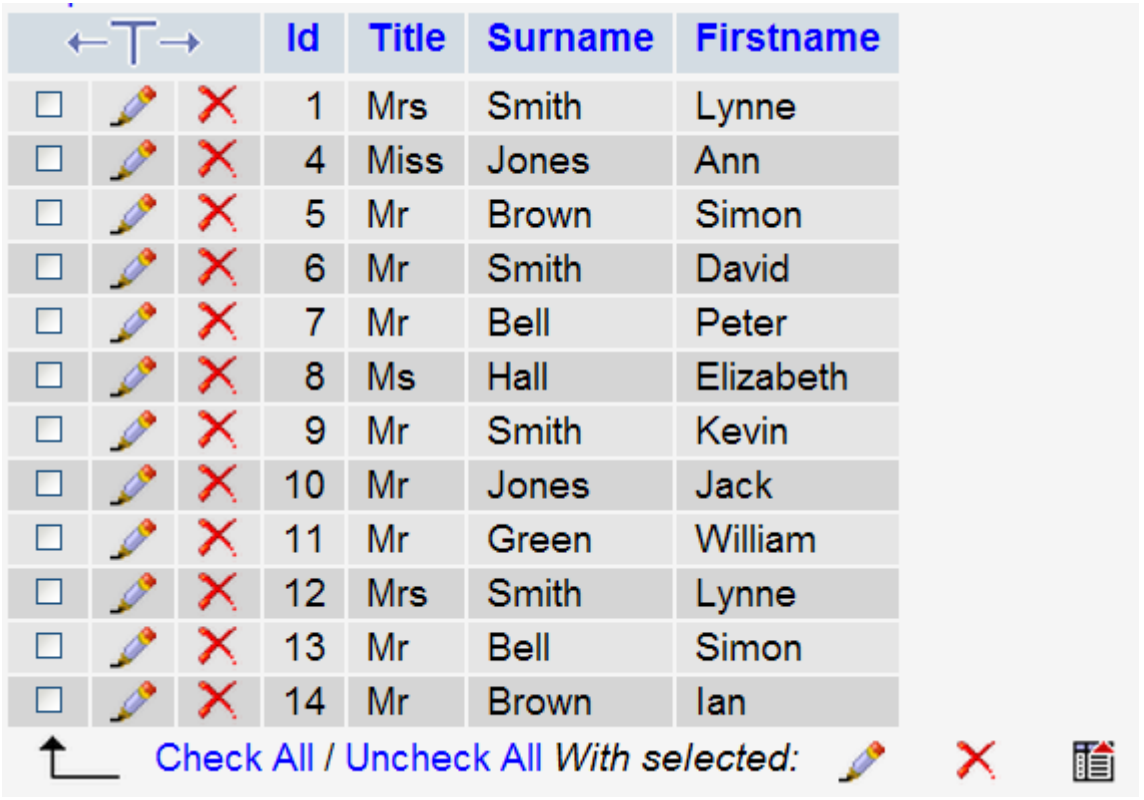
Run SQL query/queries on database glassesrus:

```
INSERT INTO `glassesrus`.`customers` (`Id`,
`Title`, `Surname`, `Firstname`) VALUES (NULL,
'Mrs', 'Smith', 'Lynne');
```

Fields

Id
Title
Surname
Firstname

Table: Customers (data)



←T→	Id	Title	Surname	Firstname
☐ ✎ ✕	1	Mrs	Smith	Lynne
☐ ✎ ✕	4	Miss	Jones	Ann
☐ ✎ ✕	5	Mr	Brown	Simon
☐ ✎ ✕	6	Mr	Smith	David
☐ ✎ ✕	7	Mr	Bell	Peter
☐ ✎ ✕	8	Ms	Hall	Elizabeth
☐ ✎ ✕	9	Mr	Smith	Kevin
☐ ✎ ✕	10	Mr	Jones	Jack
☐ ✎ ✕	11	Mr	Green	William
☐ ✎ ✕	12	Mrs	Smith	Lynne
☐ ✎ ✕	13	Mr	Bell	Simon
☐ ✎ ✕	14	Mr	Brown	Ian

Check All / Uncheck All *With selected:* ✎ ✕

Insert Record: Products

```
INSERT INTO `glassesrus`.`products` (
  `Id` ,
  `Name` ,
  `Description` ,
  `Quantity` ,
  `Cost`
)
VALUES (
  NULL , 'Beer Glass', '600 ml Beer Glass', '345', '3.99'
);
```

[Edit] [Cre

Run SQL query/queries on database glassesrus:

```
INSERT INTO `glassesrus`.`products` (`Id`, `Name`,
`Description`, `Quantity`, `Cost`) VALUES (NULL, 'Beer
Glass', '600 ml Beer Glass', '345', '3.99');
```

Fields

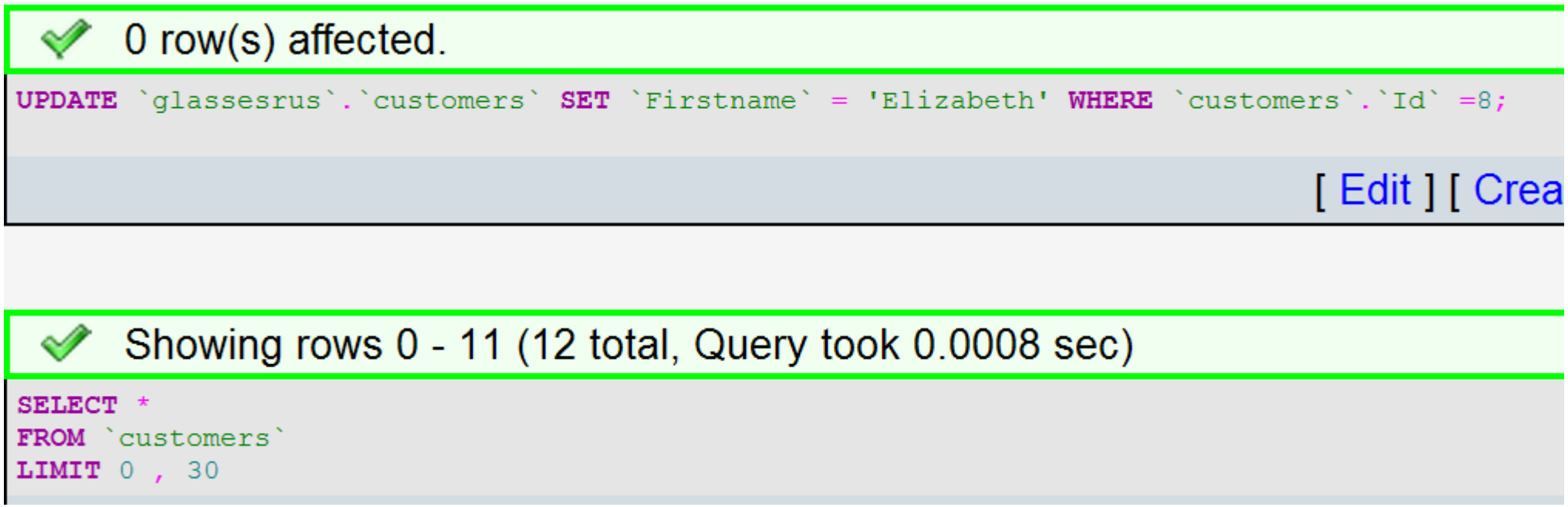
Id
Name
Description
Quantity
Cost

Table: Products (data)

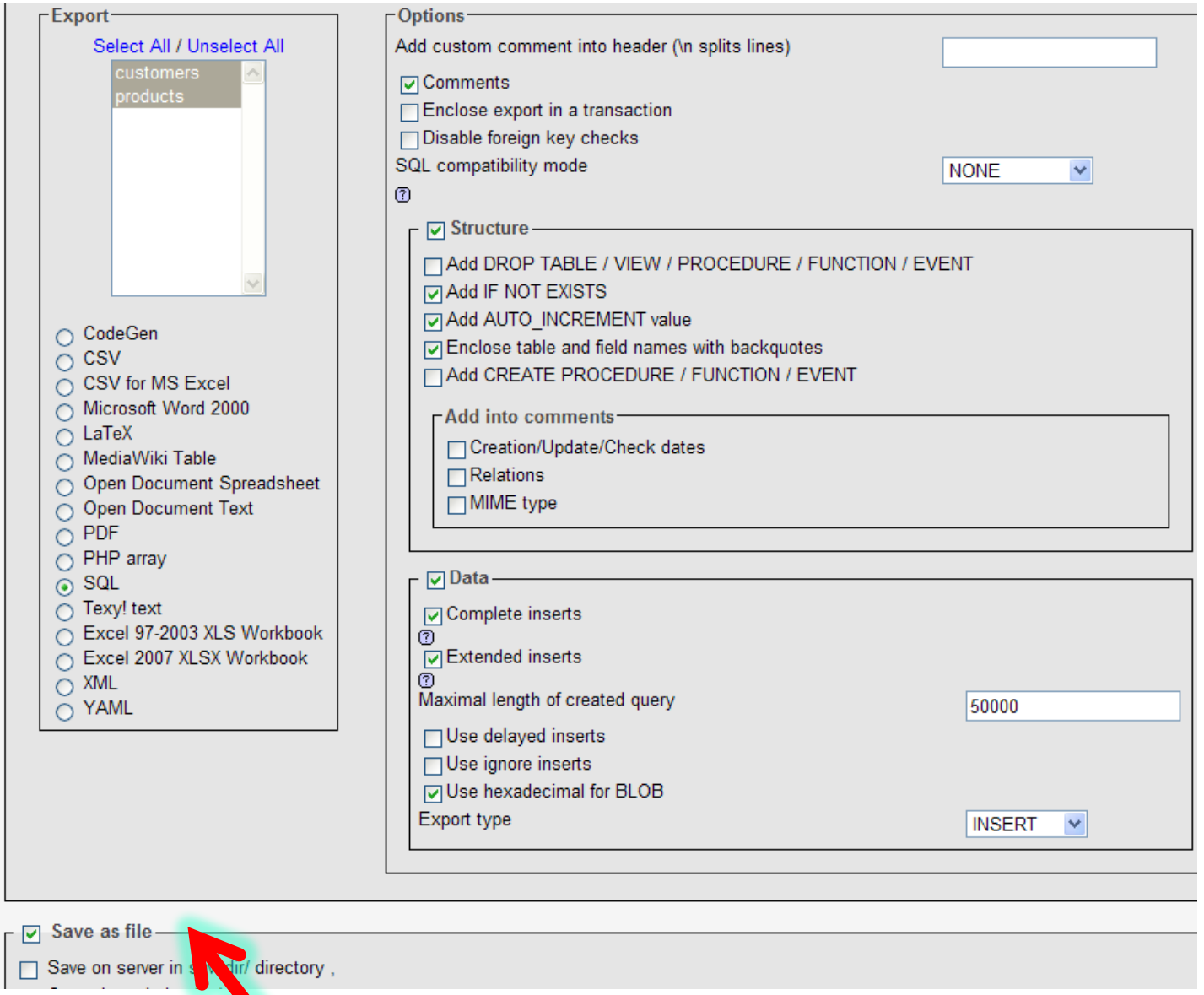
	Id	Name	Description	Quantity	Cost
☐	1	Beer Glass	600 ml Beer Glass	345	3.99
☐	2	Wine Glass	125 ml Wine Glass	236	2.99
☐	3	Wine Glass	175 ml Wine Glass	436	3.5
☐	4	Shot Glass	50 ml Small Glass	132	1.5
☐	5	Spirit Glass	100 ml Short Glass	489	2.5
☐	6	Long Glass	200 ml Tall Glass	263	2.5
☐	7	Beer Glass	300 ml Beer Glass	247	2.99
☐	8	Wine Glass	225 ml Wine Glass	96	3.99

Check All / Uncheck All *With selected:*

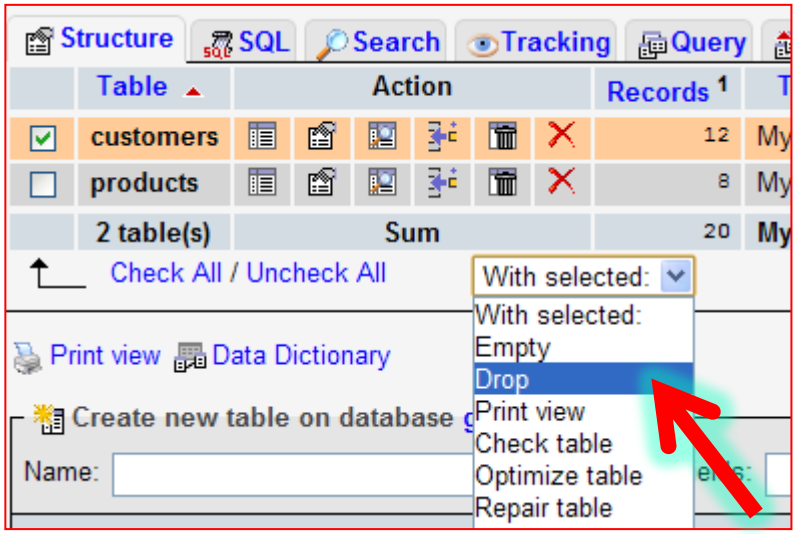
Edit Record

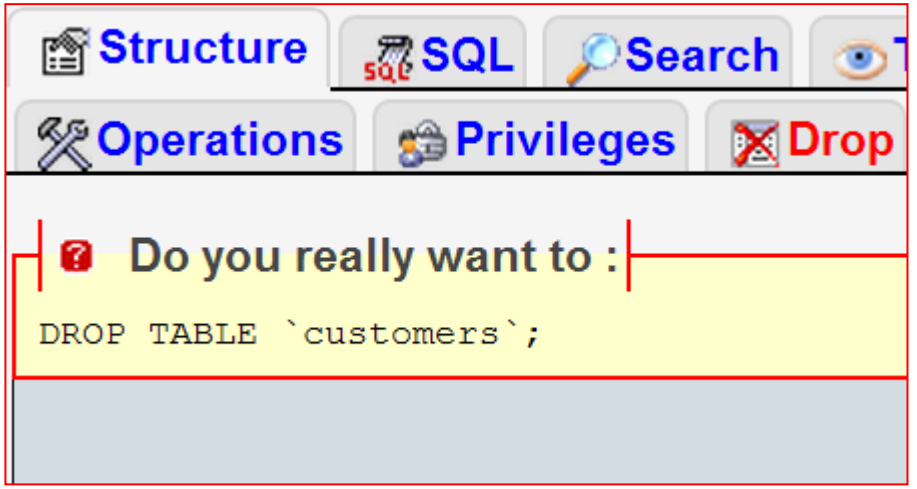

0 row(s) affected.
UPDATE `glassesrus`.`customers` SET `Firstname` = 'Elizabeth' WHERE `customers`.`Id` =8;
[Edit] [Crea
Showing rows 0 - 11 (12 total, Query took 0.0008 sec)
SELECT *
FROM `customers`
LIMIT 0 , 30

Export

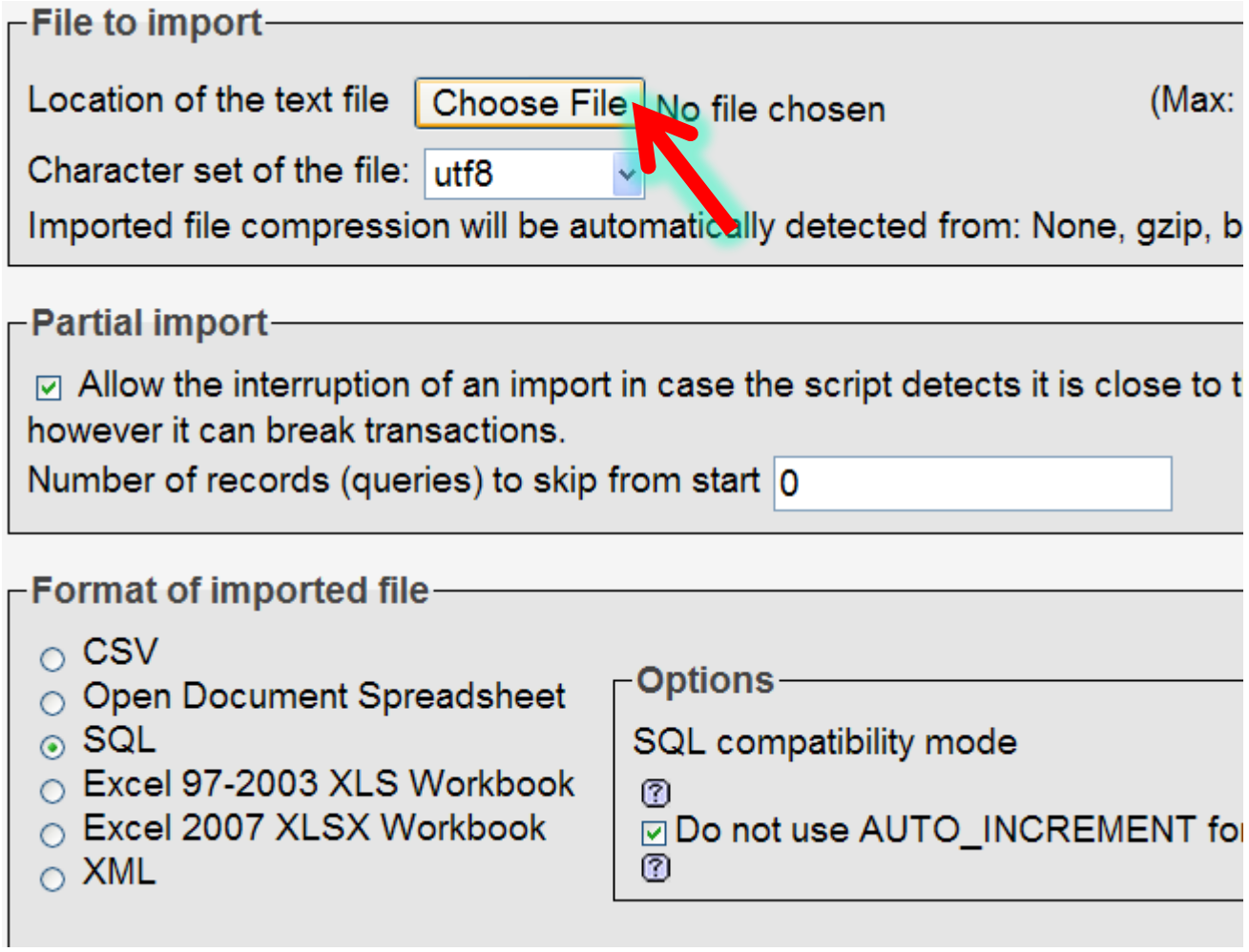

Export
Select All / Unselect All
customers
products
CodeGen
CSV
CSV for MS Excel
Microsoft Word 2000
LaTeX
MediaWiki Table
Open Document Spreadsheet
Open Document Text
PDF
PHP array
SQL
Texy! text
Excel 97-2003 XLS Workbook
Excel 2007 XLSX Workbook
XML
YAML
Options
Add custom comment into header (\n splits lines)
Comments
Enclose export in a transaction
Disable foreign key checks
SQL compatibility mode
NONE
Structure
Add DROP TABLE / VIEW / PROCEDURE / FUNCTION / EVENT
Add IF NOT EXISTS
Add AUTO_INCREMENT value
Enclose table and field names with backquotes
Add CREATE PROCEDURE / FUNCTION / EVENT
Add into comments
Creation/Update/Check dates
Relations
MIME type
Data
Complete inserts
Extended inserts
Maximal length of created query
50000
Use delayed inserts
Use ignore inserts
Use hexadecimal for BLOB
Export type
INSERT
Save as file
Save on server in directory ,

Deleting a Table

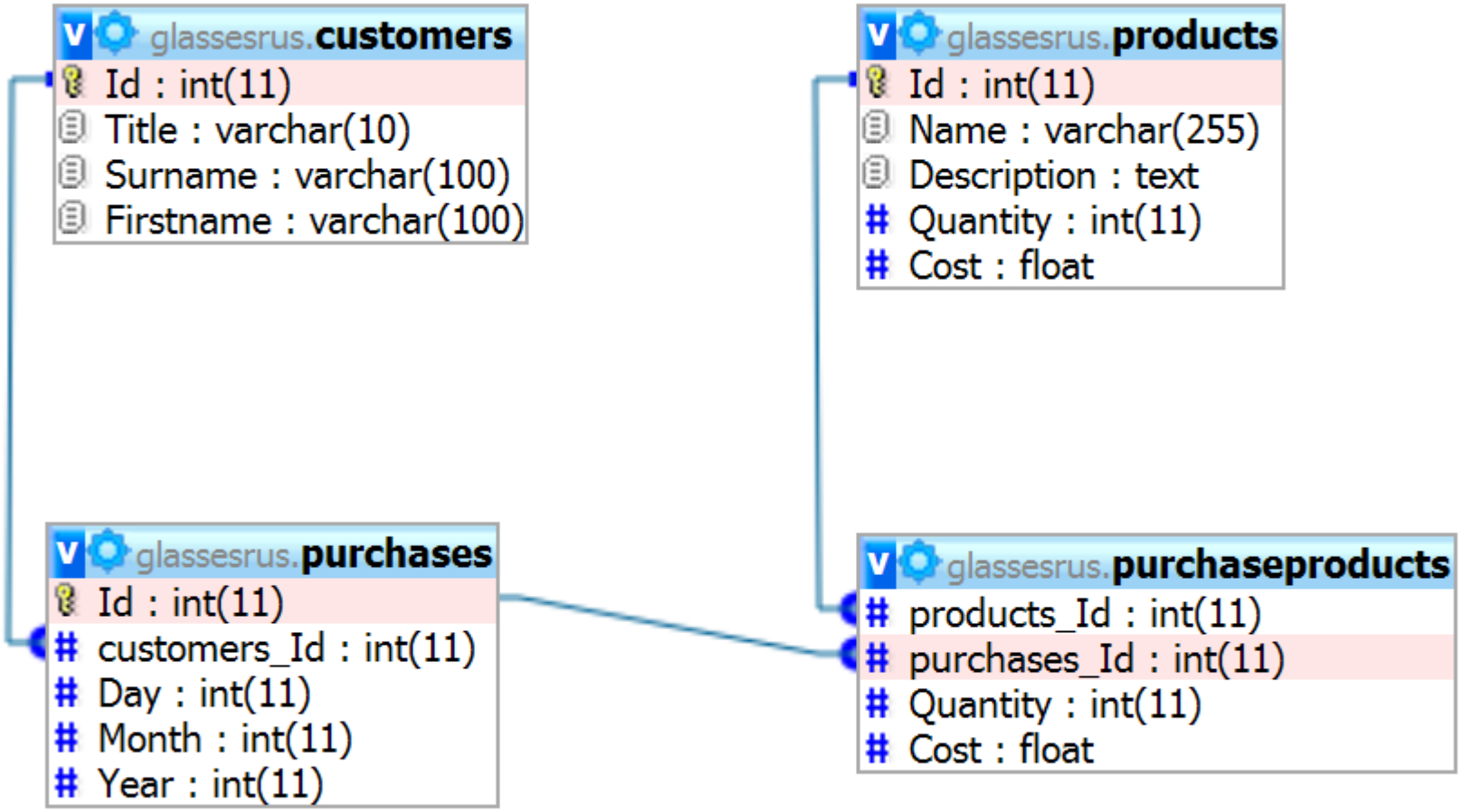




Restoring a database from an SQL file



Database Design



Summary

- Concept of databases
- Tables and Fields
- Field Types
- phpMyAdmin Tool for manipulating databases
- Creation of a database
- How to add and edit records
- How to back-up a database
- Database Design

MySQL and PHP

Connecting to a MySQL DBMS

- **In order for our PHP script to access a database we need to form a connection from the script to the database management system.**

```
resourceId = mysql_connect(server, username, password);
```

- Server is the DBMS server
- username is your username
- password is your password

Connecting to a MySQL DBMS

- **In order for our PHP script to access a database we need to form a connection from the script to the database management system.**

```
resourceId = mysql_connect(server, username, password);
```

• The function returns a resource-identifier type.
• a PHP script can connect to a DBMS anywhere in the world, so long as it is connected to the internet.
• we can also connect to multiple DBMS at the same time.

Selecting a database

- **Once connected to a DBMS, we can select a database.**

```
mysql_select_db(databasename, resourceId);
```

- the resourceId is the one returned by mysql_connect()
- the function returns true if the selection succeeded; false, otherwise.

Example: Connect to a DBMS and access database

```
<?php
$dbLocalhost = mysql_connect("localhost", "root", "")
    or die("Could not connect: " . mysql_error());
mysql_select_db("glassesrus", $dbLocalhost)
    or die("Could not find database: " . mysql_error());
echo "<h1>Connected To Database</h1>";
?>
```

- **die()** stops execution of script if the database connection attempt failed.
- **mysql_error()** returns an error message from the previous MYSQL operation.

Reading from a database

- **We can now send an SQL query to the database to retrieve some data records.**

```
resourceRecords = mysql_query(query, resourceId);
```

- the resourceId is the one returned by mysql_connect()
- the function returns a resource identifier to the returned data.

Example: Connect to a DBMS, access database, send query

```
<?php
$dbLocalhost = mysql_connect("localhost", "root", "")
    or die("Could not connect: " . mysql_error());
mysql_select_db("glassesrus", $dbLocalhost)
    or die("Could not find database: " . mysql_error());
$dbRecords = mysql_query("SELECT * FROM customers", $dbLocalhost)
   or die("Problem reading table: " . mysql_error());
echo "<h1>Connected To Database</h1>";
?>
```

- **the function will return a resource pointer (not the actual data) to all the records that match the query.**
- **If all goes well, this script will output nothing on screen.**

Extract contents of one record

- **We can now extract the actual data from the resource pointer returned by mysql_query().**

```
fieldData= mysql_result(resourceRecords, row, field);
```

- the resourceRecords is the one returned by mysql_query()
- field – database field to return
- the function returns the **data stored in the field**.

Example: Connect to a DBMS, access database, send query

```
<?php
$dbLocalhost = mysql_connect("localhost", "root", "")
    or die("Could not connect: " . mysql_error());
mysql_select_db("glassesrus", $dbLocalhost)
    or die("Could not find database: " . mysql_error());
$dbRecords = mysql_query("SELECT * FROM customers", $dbLocalhost)
   or die("Problem reading table: " . mysql_error());
$strSurname = mysql_result($dbRecords, 0, "Surname");
echo "<p>$strSurname</p>";
?>
```

- **the function will return a resource pointer (not the actual data) to all the records that match the query.**
- **If all goes well, this script will output a surname on screen.**

SQL statement

SELECT * FROM customers

- Go and **obtain** from the database
- **every** field
- **FROM** the
- **customers** table

Separating the database connection

It is worth separating the database connectivity from our scripts and placing it in a separate file.

- It provides a convenient means of moving your scripts from one database platform to another.

Example: Separating the database connection

```
<?php
// File: database2.php
$strLocation = "Home";
//$strLocation = "Work";
if ($strLocation == "Home") {
    $dbLocalhost = mysql_connect("localhost", "root", "")
          or die("Could not connect: " . mysql_error());
       mysql_select_db("glassesrus", $dbLocalhost)
       or die("Could not find database: " . mysql_error());
} else {
    $dbLocalhost = mysql_connect("localhost", "username", "password")
          or die("Could not connect: " . mysql_error());

       mysql_select_db("anotherdatabase", $dbLocalhost)
       or die("Could not find database: " . mysql_error());
}
?>
```

- **$strLocation could be easily switched between 'Home' or 'Work'**

Viewing a whole record

To view the whole record returned from mysql_query(), we need another function...

```
array = mysql_fetch_row(resourceRecords)
```

- resourceRecords – resource identifier returned from mysql_query().
- it returns an array containing the database record.

Example: Displaying all customer records

```
<?php

require_once("database2.php");

$dbRecords = mysql_query("SELECT * FROM customers", $dbLocalhost)
    or die("Problem reading table: " . mysql_error());

while ($arrRecord = mysql_fetch_row($dbRecords)) {
    echo "<p>" . $arrRecord[0] . " ";
    echo         $arrRecord[1] . " ";
    echo         $arrRecord[2] . " ";
    echo         $arrRecord[3] . "</p>";
}
?>
```

- **The function returns false when the last record is returned; thus, stopping the loop.**
- **Note, however, that the fields are referred to by using numbers – not very easy to read and mistakes can be introduced.**

Limiting the records returned

SELECT Surname FROM customers

- Retrieves only the Surname field from the table customers

Limiting the records returned

SELECT * FROM customers LIMIT 3,4

- Select a certain number of records form a table
- 3 is the starting row
- 4 is the number of records to be selected after the starting row

Searching for matching records

SELECT * FROM customers WHERE Title='Mr'

- The WHERE attribute specifies what to search for within the database records.
- in this example, only records which have a title of 'Mr' will be returned.

Searching for matching records

SELECT * FROM customers WHERE Title='Mr' OR Title='Mrs'

- The WHERE attribute specifies what to search for within the database records.
- in this example, only records which have a title of 'Mr' or 'Mrs' will be returned.
- we can also use **AND** and **OR** to formulate more sophisticated conditions.
-

Searching for matching records

SELECT * FROM customers WHERE Title='Mr' AND Surname='Smith' OR Title='Mrs'

- The WHERE attribute specifies what to search for within the database records.
- in this example, only records which have a surname of 'Smith and title of 'Mr' or the title of 'Mrs' will be returned.
- we can also use **AND** and **OR** to formulate more sophisticated conditions.
-

Sorting records

The ORDER BY attribute can be used to sort the order in which records are obtained.

```
SELECT * FROM cutomers ORDER BY Surname DESC
```

- the ORDER BY attribute is followed by the data field on which to sort the record

- DESC or ASC – from high to low, or from low to high

Example15-12.php

Accessing Multiple Tables

```
<?php
// File: example15-13.php

require_once("database2.php");

$dbRecords = mysql_query("SELECT * FROM customers WHERE Title = 'Mrs'", $dbLocalhost)
    or die("Problem reading table: " . mysql_error());

echo "<p>Customers:</p>";
while ($arrRecords = mysql_fetch_array($dbRecords)) {
    echo "<p>" . $arrRecords["Id"] . " ";
    echo $arrRecords["Title"] . " ";
    echo $arrRecords["Surname"] . " ";
    echo $arrRecords["Firstname"] . "</p>";
}

//...continued...
```

Example15-13.php

Accessing Multiple Tables

```
//continuation...

$dbRecords = mysql_query("SELECT * FROM products WHERE Name = 'Wine Glass'",
$dbLocalhost)
    or die("Problem reading table: " . mysql_error());

echo "<p>Products:</p>";
while ($arrRecords = mysql_fetch_array($dbRecords)) {
    echo "<p>" . $arrRecords["Id"] . " ";
    echo $arrRecords["Name"] . " ";
    echo $arrRecords["Description"] . " ";
    echo $arrRecords["Quantity"] . " ";
    echo $arrRecords["Cost"] . "</p>";
}
?>
```

Example15-13.php

Using records to read another table

Read a customer record, and then show the products purchased by that customer.

Tables

- Customers
- Products
- Purchases
- PurchaseProducts

Example15-14.php

BIRD'S EYEVIEW

Using records to read another table

```
...
$strSurname = "Jones";
$dbCustRecords = mysql_query("SELECT * FROM customers WHERE Surname = '$strSurname' ",...)
while ($arrCustRecords = mysql_fetch_array($dbCustRecords)) { //#1
    $intId = $arrCustRecords["Id"];
    //display customer's details
    $dbPurRecords = mysql_query("SELECT * FROM purchases WHERE customers_Id = '$intId'", ...)
while ($arrPurRecords = mysql_fetch_array($dbPurRecords)) {//#2
    $intPurId = $arrPurRecords["Id"];
    //display purchase date
$dbProRecords=mysql_query("SELECT * FROM purchaseProducts WHERE purchases_Id='$intPurId' ",..)
    while ($arrProRecords = mysql_fetch_array($dbProRecords)) { //#3
        $intProductId = $arrProRecords["products_Id"];
        //display Quantity
        $dbProductRecords = mysql_query("SELECT * FROM products WHERE Id = '$intProductId'",..)
        $arrProductRecord = mysql_fetch_array($dbProductRecords);
        //display product details
     } #3
   } #2
} //#1
```

Example15-14.php

Complete version

Using records to read another table

```
<?php
require_once("database2.php");

$strSurname = "Jones";

$dbCustRecords = mysql_query("SELECT * FROM customers WHERE Surname = '$strSurname'
", $dbLocalhost)
    or die("Problem reading table: " . mysql_error());

while ($arrCustRecords = mysql_fetch_array($dbCustRecords)) {
    $intId = $arrCustRecords["Id"];
    echo "<p>Customer: ";
    echo $arrCustRecords["Title"] . " ";
    echo $arrCustRecords["Surname"] . " ";
    echo $arrCustRecords["Firstname"] . "</p>";

    $dbPurRecords = mysql_query("SELECT * FROM purchases WHERE customers_Id = '$intId'",
$dbLocalhost)
    or die("Problem reading table: " . mysql_error());
```

Example15-14.php

Complete version

Using records to read another table

```
while ($arrPurRecords = mysql_fetch_array($dbPurRecords)) {
        $intPurId = $arrPurRecords["Id"];
        echo "<p>Purchased On: ";
        echo $arrPurRecords["Day"] . "/";
        echo $arrPurRecords["Month"] . "/";
        echo $arrPurRecords["Year"] . "</p>";

     $dbProRecords= mysql_query("SELECT * FROM purchaseProducts WHERE purchases_Id='$intPurId' ",
$dbLocalhost)
    or die("Problem reading table: " . mysql_error());

    while ($arrProRecords = mysql_fetch_array($dbProRecords)) {
        $intProductId = $arrProRecords["products_Id"];
        echo "<p>" . $arrProRecords["Quantity"] . " ";

        $dbProductRecords = mysql_query("SELECT * FROM products WHERE Id = '$intProductId'",
$dbLocalhost)
    or die("Problem reading table: " . mysql_error());

        $arrProductRecord = mysql_fetch_array($dbProductRecords);
        echo $arrProductRecord["Name"] . " (" . $arrProductRecord["Description"] . ") at &#163;";
        echo $arrProRecords["Cost"] . " each.</p>";
        }
    }
}
?>
```

Example15-14.php

Inserting records

How to create new database records and insert them into a table?

INSERT INTO table (field1, field2,...) VALUES ('value1', 'value2',...)

- Alternatively, we have a simplified syntax:

INSERT INTO table VALUES ('value1', 'value2',...)

```
$dbProdRecords = mysql_query("INSERT INTO products
VALUES ( ' ', 'Beer Mug', '600 ml Beer Mug', '100', '5.99')",
$dbLocalhost)
```

Example15-15.php

Inserting records

```
<?php
// File: example15-15.php

require_once("database2.php");

$dbProdRecords = mysql_query("INSERT INTO products VALUES ('', 'Beer Mug', '600
ml Beer Mug', '100', '5.99')", $dbLocalhost)
   or die("Problem writing to table: " . mysql_error());

$dbProdRecords = mysql_query("SELECT * FROM products", $dbLocalhost)
    or die("Problem reading table: " . mysql_error());

while ($arrProdRecords = mysql_fetch_array($dbProdRecords)) {
    echo "<p>" . $arrProdRecords["Id"] . " ";
    echo $arrProdRecords["Name"] . " ";
    echo $arrProdRecords["Description"] . " ";
    echo $arrProdRecords["Quantity"] . " ";
    echo $arrProdRecords["Cost"] . "</p>";
}
?>
```

Example15-15.php

Deleting records

How to delete database records from tables?

DELETE FROM table WHERE field='value'

e.g.

```
$dbCustRecords = mysql_query("DELETE FROM customers
WHERE Id='3'", $dbLocalhost)
```

Note: If you have a relational database, you should tidy-up the other tables, based on their connection with the record you've deleted.

Example15-16.php

Deleting records

How to delete database records from tables?

DELETE FROM table

This will delete all records from a table!

Note: back-up your database first!

Example15-17.php

Amending records

How to modify the contents of an existing database record?

UPDATE table SET field='value1', field='value2'...WHERE field='value'

• requires you to specify the table, the list of fields with their updated values, and a condition for selection (WHERE).

Example15-18.php

Amending records

```
<?php
// File: example15-18.php

require_once("database2.php");

$dbCustRecords = mysql_query("UPDATE products SET Description='250 ml Tall
Glass' WHERE Id='6'", $dbLocalhost)
   or die("Problem updating table: " . mysql_error());

$dbProdRecords = mysql_query("SELECT * FROM products", $dbLocalhost)
    or die("Problem reading table: " . mysql_error());

while ($arrProdRecords = mysql_fetch_array($dbProdRecords)) {
    echo "<p>" . $arrProdRecords["Id"] . " ";
    echo $arrProdRecords["Name"] . " ";
    echo $arrProdRecords["Description"] . " ";
    echo $arrProdRecords["Quantity"] . " ";
    echo $arrProdRecords["Cost"] . "</p>";
}
?>
```

Example15-18.php

Amending records

How to modify the contents of an existing database record?

UPDATE table SET field='value1', field='value2'...WHERE field='value'

Another Example:

```
$dbCustRecords = mysql_query("UPDATE products SET Name='Beer and Lager Glass' WHERE Name='Beer Glass'", $dbLocalhost)
```

- A number of records will be updated in this example.

Example15-19.php

Counting the number of records

How to count the number of records after running a query?

```
$dbProdRecords = mysql_query("SELECT * FROM products",
    $dbLocalhost)

    or die("Problem reading table: " . mysql_error());


$intProductCount = mysql_num_rows($dbProdRecords);
```

- you can also use the same function to determine if a record exists.

Example15-20.php

Example15-21.php

Select a substring

How to count the number of records after running a query?

```
SELECT * FROM products WHERE substring(Name,1,4)='Wine'
```

•This will return all records from the products table where the first four characters in the name field equals 'Wine'

Example15-22.php

End of Lecture