



MYSQL

By Prof. Vaibhavi Patel



Content

- Introduction about Database
- Data Types
- DDL, DML
- Aggregate functions
- Date Time functions
- Sub query and join
- MySQL Introduction, MySQL Connect, MySQL Create, MySQL Insert, MySQL Select, MySQL Where, MySQL Order By, MySQL Update, MySQL Delete, phpMyAdmin.



Introduction

To build a web site that shows data from a database, you will need:

- An RDBMS database program (i.e. MS Access, SQL Server, MySQL)
- To use a server-side scripting language, like PHP or ASP
- To use SQL to get the data you want
- To use HTML / CSS to style the page



Introduction about Database

➤ Data

- Fact that can be recorded or stored.
- E.g. Person Name, Age, Gender and Weight etc.

➤ Information

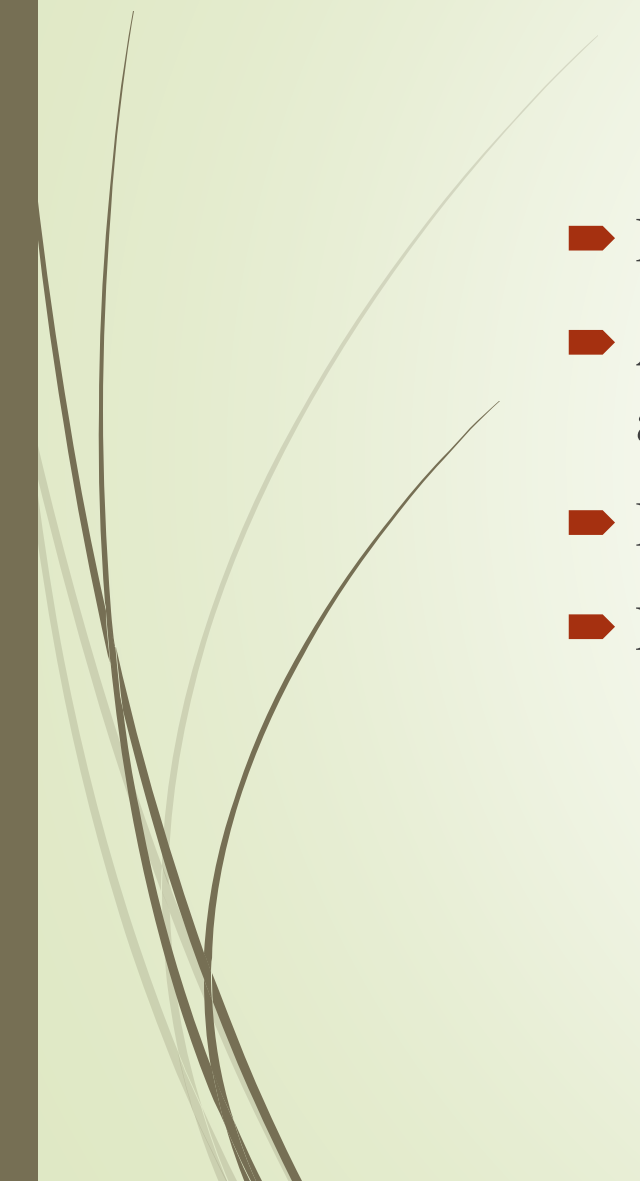
- When data is processed, organized, structured or presented in a given context so as to make it useful, it is called information.

➤ Database

- A Database is a collection of inter-related data.
- E.g. Books Database in Library, Student Database in University etc.



Introduction about Database

- DBMS (Database Management System)
 - A database management system is a collection of inter-related data and set of programs to manipulate those data.
 - DBMS = Database + Set of programs
 - E.g. MS SQL Server, Oracle, My SQL, SQLite, MongoDB etc
- 

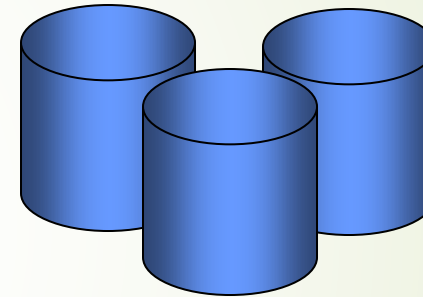
Structure of a Database

- A database system may contain many databases.
- Each database is composed tables.

```
sql> SHOW databases;
```

+-----+	
Database	
+-----+	
mysql	
test	
bank	
world	
+-----+	

MySQL only shows databases that a user has permission to access.



```
sql> USE bank;
```

```
sql> SHOW tables;
```

+-----+	
Tables_in_bank	
+-----+	
accounts	
clients	
+-----+	

Contents of a Table

- A table contains the actual data in **records** (rows).
- A record is composed of **fields** (columns).
- Each record contains one set of data values.

records
(rows)

ID	Name	CCode	District	Populatn
3320	Bangkok	THA	Bangkok	6320174
3321	Nonthaburi	THA	Nonthaburi	292100
3323	Chiang Mai	THA	Chiang Mai	171100

fields (columns)

Key field for Identifying Rows

- A table contains a **primary key** that uniquely identifies a row of data.
- Each record must have a distinct value of primary key
- The primary key is used to relate (join) tables.

ID is the primary key in City table.

ID	Name	CCode	District	Populatn
3320	Bangkok	THA	Bangkok	6320174
3321	Nonthaburi	THA	Nonthaburi	292100
3323	Chiang Mai	THA	Chiang Mai	171100

Structure of a Table

Every field has:

- ▶ a name
- ▶ a data type and length

To view the structure of a table use:

DESCRIBE tablename;

Fields may have a default value to use if a value is not assigned explicitly.

```
sql> DESCRIBE City;
```

Field	Type	Null	Key	Default	Extra
ID	int(11)	NO	PRI		auto_increment
Name	char(35)	NO			
CountryCode	char(3)	NO			
District	char(20)	NO			
Population	int(11)	NO		0	



Data Types

- Each column in a database table is required to have a name and a data type.
- An SQL developer must decide what type of data that will be stored inside each column when creating a table.



Data Types

Data type	Description
CHAR(size)	A FIXED length string. The size can be from 0 to 255. Default is 1
VARCHAR(size)	A VARIABLE length string. The size can be from 0 to 65535
INT(size) or INTEGER(size)	A medium integer
TINYINT(size)	A very small integer
BIGINT(size)	A large integer
FLOAT(size)	A floating point number. The total number of digits is specified in size.
DATE	A date. Format: YYYY-MM-DD

Structured Query Language

- **Structured Query Language (SQL)** is the standard language for accessing information a database.
- SQL is case-insensitive and free format.
- SQL statements can use multiple lines
 - end each statement with a semi-colon ;

```
sql> USE world;
```

```
database changed.
```

```
sql> SHOW tables;
```

```
sql> SHOW columns FROM city;
```

```
sql> DESCRIBE country;
```

SQL statements end with **semi-colon**.



Structured Query Language

➤ It can be categorized in 4 types.

1. DDL
2. DML
3. DCL
4. DQL



Structured Query Language - DDL

DDL (Data Definition Language)

- It is a set of SQL commands used to create, modify and delete database objects such as tables.
- It is normally used by DBA.
- It provides commands like:
 - CREATE: to create tables in a database.
 - ALTER: to alter the schema.
 - DROP: to delete tables from the database.
 - TRUNCATE: to remove all records from the table.

Structured Query Language - DDL

Create Table

The CREATE TABLE statement is used to create a new table in a database.

Syntax:

```
CREATE TABLE table_name  
(  
  Column1 Datatype(Size) [ NULL | NOT NULL ],  
  Column2 Datatype(Size) [ NULL | NOT NULL ],  
  ...  
);
```

Example:

```
CREATE TABLE Students  
(  
  Roll_No int(3) PRIMARY KEY,  
  Name varchar(20),  
  Subject varchar(20) NOT NULL ,  
  Marks int(20) CHECK (marks >= 0),  
  Contact_No int(10) UNIQUE NOT NULL  
);
```


Structured Query Language - DDL

ALTER: ALTER TABLE statement is used to add, modify, or drop columns in a table.

➤ Add Column

The ALTER TABLE statement in SQL to add new columns in a table.

Syntax:

```
ALTER TABLE table_name  
ADD Column1 Datatype(Size) ;
```

Example:

```
ALTER TABLE Students  
ADD Marks int;
```

Structured Query Language - DDL

ALTER: ALTER TABLE statement is used to add, modify, or drop columns in a table.

➤ Drop Column

The ALTER TABLE statement in SQL to drop a column in a table.

Syntax:

```
ALTER TABLE table_name  
DROP column_name;
```

Example:

```
ALTER TABLE Students  
DROP Subject;
```

Structured Query Language - DDL

ALTER: ALTER TABLE statement is used to add, modify, or drop columns in a table.

➤ Modify Column

The ALTER TABLE statement in SQL to change the data type/size of a column in a table.

Syntax:

```
ALTER TABLE table_name  
MODIFY column_name datatype(size);
```

Example:

```
ALTER TABLE Students MODIFY Roll_No  
float;
```

Structured Query Language - DDL

ALTER: ALTER TABLE statement is used to rename a table.

- Modify table name

The ALTER TABLE statement in SQL to change the table name in a table.

Syntax:

```
ALTER TABLE table_name  
Rename TO new_table_name;
```

Example:

```
ALTER TABLE Students rename to student;
```

Structured Query Language - DDL

ALTER: ALTER TABLE statement is used to rename a column.

- Modify column name

The ALTER TABLE statement in SQL to change the column name in a table.

Syntax:

```
ALTER TABLE table_name  
change column original_name new_name  
column_definition;
```

Example:

```
ALTER TABLE Students change Roll_No  
RollNo int(10);
```

Structured Query Language - DDL

DROP: Drop is used to drop the database or its objects like table.

➤ Drop Table

The DROP TABLE statement is used to drop an existing table in a database.

Syntax:

```
DROP TABLE table_name;
```

Example:

```
DROP TABLE Students;
```



Structured Query Language - DDL

TRUNCATE: Truncate is used to remove all records from the table

Syntax:

```
TRUNCATE TABLE table_name;
```

Example:

```
TRUNCATE TABLE Students;
```




Structured Query Language - DML

- DML (Data Manipulation Language)
- It is a set of SQL commands used to insert, modify and delete data in a database.
- It is normally used by general users who are accessing database via pre-developed applications.
- It provides commands like:
 - INSERT: to insert data into a table.
 - UPDATE: to modify existing data in a table.
 - DELETE: to delete records from a table.

Structured Query Language - DML

INSERT: The INSERT STATEMENT is used to insert data into a table

Syntax:

```
INSERT INTO table_name (column1, column2,  
column3, ...)  
VALUES (value1, value2, value3, ...);
```

OR

```
INSERT INTO table_name  
VALUES (value1, value2, value3, ...);
```

Example:

```
INSERT INTO Students VALUES  
(1,'anil','Maths',12,123);  
OR
```

```
INSERT INTO Students (Roll_No, Name, Subject,  
contact_no)  
VALUES (1,'anil','Maths',123);
```

Structured Query Language - DML

UPDATE: The UPDATE STATEMENT is used to modify existing data in a table

Syntax:

```
UPDATE table_name  
SET column1 = value1, column2 = value2, ...  
WHERE condition;
```

Example:

```
UPDATE Students SET Name= 'Mahesh'  
WHERE Roll_No=1;
```

Structured Query Language - DML

DELETE: The DELETE STATEMENT is used to delete records from a table.

Syntax:

```
DELETE FROM table_name WHERE condition;
```

Example:

```
DELETE FROM Students WHERE  
Subject='Maths';
```



Structured Query Language - DQL

DQL (Data Query Language):

- It is a component of SQL that allows data retrieval from the database.
- It provides command like SELECT. This command is a heart of SQL, and allows data retrieval in different ways

Structured Query Language - DQL

SELECT: The SELECT statement is used to select data from a database. The data returned is stored in a result table, called the result-set.

Syntax:

```
SELECT column1, column2, ...  
FROM table_name  
WHERE condition;
```

OR

```
SELECT *  
FROM table_name  
WHERE condition;
```

Example:

```
SELECT *  
FROM Students  
WHERE Roll_No >= 1;
```



Structured Query Language – copy table

The CREATE TABLE statement is used to create a new table (copy of old table) in a database.

Example:

```
create table depositnew as select * from deposit;
```

- It will create copy table of deposit



Structured Query Language - DCL

DCL (Data Control Language)

- It is set of SQL commands used to control access to data and database. Occasionally DCL commands are grouped with DML commands.

It provides commands like:

- GRANT: to give access privileges to users on the database.
- REVOKE: to withdraw access privileges given to users on the database



Aggregate functions

- Aggregate functions perform a calculation on a set of values and return a single value.
- Aggregate functions ignore NULL values except COUNT(*).
- It is used with the GROUP BY clause of the SELECT statement
 - For ex: >> select max(marks), subject FROM Students group by(subject);



Aggregate functions

- Avg() : It returns the average of the data values.

Select Avg(marks) FROM Students;


Output: 7

- Sum() : It returns the addition of the data values
- Max() : It returns maximum value for a column.
- Min() : It returns Minimum value for a column.
- Count() : It returns total number of values in a given column
- COUNT(*): It returns total number of rows in a given table



Date Time functions

- `CURDATE()` // Returns the current date as a value in 'YYYY-MM-DD'
- `CURTIME()` //Returns the current time as a value in 'HH:MM:SS'
- `SYSDATE()` // Returns the current date and time as a value in 'YYYY-MM-DD HH:MM:SS'
- `DATEDIFF(date1,date2)` // days between date1 and date2
- `DAYNAME(date)` // Returns the name of the weekday for date
- `DAYOFMONTH(date)` //Returns the day of the month for date, in the range 0 to 31
- `DAYOFWEEK(date)` //Returns the weekday index for date (1 = Sunday, 2 = Monday, ., 7 = Saturday)
- `STR_TO_DATE('date1','format of date1')` // convert date1 into system date format



Sub query

- A MySQL subquery is a query nested within another query.
- A MySQL subquery is called an inner query while the query that contains the subquery is called an outer query
- Example : the following query returns the students who has the maximum marks.

```
SELECT * FROM students  
WHERE  
marks = (SELECT MAX(marks) FROM students);
```



ORDER BY Keyword

- The ORDER BY keyword is used to sort the result-set in ascending or descending order.
- The ORDER BY keyword sorts the records in ascending order by default.
- To sort the records in descending order, use the DESC keyword.

Syntax:

```
SELECT column1, column2, ...  
FROM table_name  
ORDER BY column_name ASC|DESC
```



Join

A SQL Join statement is used to combine data or rows from two or more tables based on a common field between them.

Different types of Joins are:

- Cross join
- Inner Join
- Outer Join
 - 1. Left Outer Join
 - 2. Right Outer Join
- Self Join

Cross join: When each row of first table is combined with each row from the second table, known as Cartesian join or cross join.

Syntax:

```
SELECT columns  
FROM table1 CROSS JOIN table2;
```

Example:

```
SELECT Color.Code, Color.Name,  
Size.Amount  
FROM Color CROSS JOIN Size;
```

Example:

Consider the following tables

Color		Size
Code	Name	Amount
1	Red	Small
2	Blue	Large



Output:

Cross Join		
Code	Name	Amount
1	Red	Small
2	Blue	Small
1	Red	Large
2	Blue	Large

Inner join: It returns records that have matching values in both tables.

Syntax:

```
SELECT columns  
FROM table1 INNER JOIN table2  
ON table1.column = table2.column;
```

Example:

```
SELECT Student.RNO, Student.Name,  
Student.Branch, Result.SPI  
FROM Student INNER JOIN Result  
ON Student.RNO = Result.RNO;
```

Example:

Consider the following tables

Student		
RNO	Name	Branch
101	Raju	CE
102	Amit	CE
103	Sanjay	ME
104	Neha	EC
105	Meera	EE
106	Mahesh	ME

Result	
RNO	SPI
101	8.8
102	9.2
104	8.2
105	7
107	8.9



Output:

Inner Join			
RNO	Name	Branch	SPI
101	Raju	CE	8.8
102	Amit	CE	9.2
104	Neha	EC	8.2
105	Meera	EE	7

Left outer join: The LEFT OUTER JOIN keyword returns all records from the left table (table1), and the matched records from the right table (table2). The result is NULL from the right side, if there is no match

Syntax:

```
SELECT columns  
FROM table1 LEFT OUTER JOIN table2  
ON table1.column = table2.column;
```

Example:

```
SELECT Student.RNO, Student.Name,  
Student.Branch, Result.SPI  
FROM Student LEFT OUTER JOIN Result  
ON Student.RNO = Result.RNO;
```

Example:

Consider the following tables

Student		
RNO	Name	Branch
101	Raju	CE
102	Amit	CE
103	Sanjay	ME
104	Neha	EC
105	Meera	EE
106	Mahesh	ME

Result	
RNO	SPI
101	8.8
102	9.2
104	8.2
105	7
107	8.9



Output:

Left Join			
RNO	Name	Branch	SPI
101	Raju	CE	8.8
102	Amit	CE	9.2
103	Sanjay	ME	NULL
104	Neha	EC	8.2
105	Meera	EE	7
106	Mahesh	ME	NULL

Right outer join: The RIGHT JOIN keyword returns all records from the right table (table2), and the matched records from the left table (table1). The result is NULL from the left side, if there is no match.

Syntax:

```
SELECT columns  
FROM table1 RIGHT OUTER JOIN table2  
ON table1.column = table2.column;
```

Example:

```
SELECT Student.RNO, Student.Name,  
Student.Branch, Result.SPI  
FROM Student RIGHT OUTER JOIN Result  
ON Student.RNO = Result.RNO;
```

Example:

Consider the following tables

Student		
RNO	Name	Branch
101	Raju	CE
102	Amit	CE
103	Sanjay	ME
104	Neha	EC
105	Meera	EE
106	Mahesh	ME

Result	
RNO	SPI
101	8.8
102	9.2
104	8.2
105	7
107	8.9



Output:

Right Join			
RNO	Name	Branch	SPI
101	Raju	CE	8.8
102	Amit	CE	9.2
104	Neha	EC	8.2
105	Meera	EE	7
NULL	NULL	NULL	8.9

Self join: A self join is a regular join, but the table is joined with itself.
Here, we need to use aliases for the same table to set a self join between single table.

Syntax:

```
SELECT a.column, b.column  
FROM tablename a INNER JOIN tablename b  
ON a.column=b.column;
```

Example:

```
SELECT e.Name as Employee, m.Name as Manager  
FROM Employee e INNER JOIN Employee m  
ON e.MngrNo=m.EmpNo;
```

Example:

Consider the following tables

Employee		
EmpNo	Name	MngrNo
E01	Tarun	E02
E02	Rohan	E05
E03	Priya	E04
E04	Milan	NULL
E05	Jay	NULL
E06	Anjana	E03



Output:

Employee	
Employee	Manager
Tarun	Rohan
Rohan	Jay
Priya	Milan
Anjana	Priya



MySQL

MySQL is a very popular, open source database.

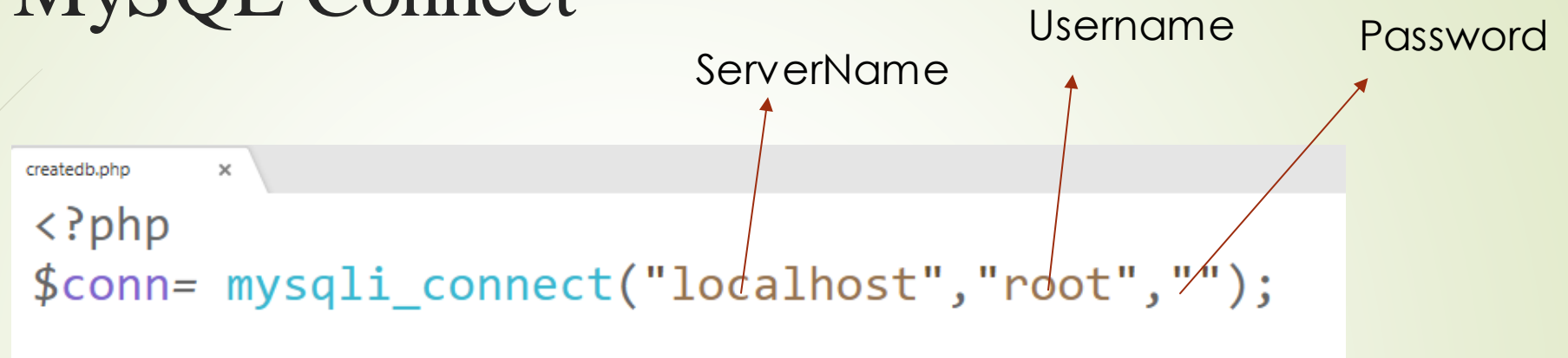


MySQL Connect

- PHP 5 and later can work with a MySQL database using MySQLi extension (the "i" stands for improved)
- *Open a Connection to MySQL*
- Before we can access data in the MySQL database, we need to be able to connect to the server
- *Method:*

```
$conn= mysqli_connect($servername, $username, $password);
```


MySQL Connect



Parameter	Description
server	Specifies the server to connect
user	Specifies the username to log in with.
pwd	Specifies the password to log in with.



Close the Connection

- The connection will be closed automatically when the script ends.
- To close the connection before, use the following:

`mysqli_close($conn)`

Create a MySQL Database

- The following example create a database named "myDB"

ServerName Username Password

```
createdb.php x
<?php
$conn= mysqli_connect("localhost","root","");

mysqli_query($conn,"CREATE DATABASE myDB") ;

mysqli_close($conn);
?>
```

Create MySQL Tables

- The following example create a table named "Mytable", with three columns: "id", "firstname", "lastname"

Database name

```
createdb.php x create_table.php • insertdata.php x
<?php

$conn= mysqli_connect("localhost","root","","myDB");

$sql = "CREATE TABLE Mytable (id int(6) AUTO_INCREMENT PRIMARY KEY,
                                firstname VARCHAR(30),
                                lastname varchar(30))";

mysqli_query($conn,$sql) ;

mysqli_close($conn);
?>
```

Insert Data Into MySQL

- After a database and a table have been created, we can start adding data in them.
- We can also insert data from HTML form when user submit the data.

```
createdb.php x create_table.php x insertdata.php x  
  
<?php  
  
$conn= mysqli_connect("localhost","root","", "myDB");  
  
$sql = "INSERT INTO Mytable (firstname,lastname) VALUES ('vaibhavi','patel')";  
  
mysqli_query($conn,$sql);  
  
mysqli_close($conn);  
?>
```



Select Data From MySQL

- The SELECT statement is used to select data from tables:

SELECT column_name(s) FROM table_name

- we can use the * character to select ALL columns from a table:

*SELECT * FROM table_name*



```
createdb.php x create_table.php x insertdata.php x sqldemo.php x
<?php


$conn= mysqli_connect("localhost","root","", "myDB");

$result = mysqli_query($conn, "SELECT id, firstname ,lastname FROM Mytable");

if (mysqli_num_rows($result) > 0) {

    while($row = mysqli_fetch_assoc($result)) {
        echo $row['id'] , $row['firstname'] , $row['lastname'] ;
    }
} else {
    echo "0 results";
}
```

- **mysqli_num_rows()** checks if there are more than zero rows returned.
- If there are more than zero rows returned, the function **mysqli_fetch_assoc()** puts all the results into an associative array that we can loop through.
- The **while()** loop loops through the result set and outputs the data from the id, firstname and lastname columns.



mysqli_fetch_assoc(data1)

- The mysqli_fetch_assoc() function returns a row from a recordset as an associative array .
- This function gets a row from the mysqli_query() function and returns an array on success, or FALSE on failure or when there are no more rows.
- data1 Specifies which data pointer to use. The data pointer is the result from the mysqli_query() function

Delete Data From MySQL

- The DELETE statement is used to delete records from a table:

`DELETE FROM table_name WHERE some_column = some_value;`

The following examples delete the record with id=3 in the "Mytable" table:

```
createdb.php x create_table.php x insertdata.php x detetetable.php x sqldemo.php x
<?php

$conn= mysqli_connect("localhost","root","", "myDB");

$sql = "DELETE FROM Mytabel WHERE id=3";

mysqli_query($conn,$sql);

mysqli_close($conn);
?>
```

By Vaibhavi Patel

Update Data in MySQL

- The UPDATE statement is used to update existing records in a table:

UPDATE table_name SET column1=value, column2=value2,... WHERE some_column=some_value ;

- The following examples update the record with id=2 in the "Mytable" table:

```
createdb.php x create_table.php x insertdata.php x detetetable.php x update_table.php x sqlc
<?php

$conn= mysqli_connect("localhost","root","", "myDB");

$sql = "UPDATE Mytable SET lastname='patel' WHERE id=2";

mysqli_query($conn,$sql);

mysqli_close($conn);
?>
```

By Vaibhavi Patel

phpMyAdmin.

The screenshot displays the phpMyAdmin web interface in a browser window. The address bar shows the URL `localhost/phpmyadmin/`. The interface includes a top navigation bar with tabs for `Databases`, `SQL`, `Status`, `User accounts`, `Export`, `Import`, `Settings`, `Replication`, `Variables`, and `More`. On the left, a sidebar lists the database structure, including a `New` button and a tree view with databases like `hem`, `information_schema`, `mydb`, `mysql`, `performance_schema`, `phpmyadmin`, `temp`, and `test`. The main content area is divided into three panels:
1. **General settings**: Shows `Server connection collation` set to `utf8mb4_unicode_ci` with a `More settings` link.
2. **Appearance settings**: Shows `Language` set to `English` and `Theme` set to `pmahomme`.
3. **Database server**: A list of server details including `Server: 127.0.0.1 via TCP/IP`, `Server type: MariaDB`, `Server connection: SSL is not being used`, `Server version: 10.4.18-MariaDB - mariadb.org binary distribution`, `Protocol version: 10`, `User: root@localhost`, and `Server charset: UTF-8 Unicode (utf8mb4)`.
4. **Web server**: A list of web server details including `Apache/2.4.46 (Win64) OpenSSL/1.1.1j PHP/7.3.27`, `Database client version: libmysql - mysqlnd 5.0.12-dev - 20150407 - $Id: 7cc7cc96e675f6d72e5cf0f267f48e167c2abb23 $`, `PHP extension: mysqli curl mbstring`, and `PHP version: 7.3.27`.
At the bottom left, there is a `Console` tab.



Thank you