MySQL

Prof. Jayakumar Sadhasivam,
School of Computer Science and Engineering,
Vellore Institute of Technology, Vellore.

MySQL Introduction

- MySQL is the most popular open source database server.
- MySQL is a database management system
- Free SQL (Structured Query Language) database server.
- Licensed with the GNU General public license http://www.gnu.org/

What is a Database?

- A database is a separate application that stores a collection of data.
- Each database has one or more distinct APIs for creating, accessing, managing, searching and replicating the data it holds.
- Other kinds of data stores can also be used, such as files on the file system or large hash tables in memory but data fetching and writing would not be so fast and easy with those type of systems.
- Database Queries:
  - ✓ A query is a question or a request.
  - ✓ We can query a database for specific information and have a recordset returned.

RDBMS

- We use Relational Database Management Systems (RDBMS) to store and manage huge volume of data.
- This is called relational database because all the data is stored into different tables and relations are established using primary keys or other keys known as foreign keys

RDBMS Terminology

- Database: A database is a collection of tables, with related data.
- Table: A table is a matrix with data. A table in a database looks like a simple spreadsheet.
- Column: One column (data element) contains data of one and the same kind, for example the column postcode.
- Row: A row (=tuple, entry or record) is a group of related data, for example the data of one subscription.
- Primary Key: A primary key is unique. A key value can not occur twice in one table. With a key you can find at most one row.
- Foreign Key: A foreign key is the linking pin between two tables.
- Index: An index in a database resembles an index at the back of a book.

What is MySQL?

- MySQL is a database system used on the web
- MySQL is a database system that runs on a server
- MySQL is ideal for both small and large applications
- MySQL is very fast, reliable, and easy to use
- MySQL uses standard SQL
- MySQL compiles on a number of platforms
- MySQL is free to download and use
- MySQL was a Swedish software company founded in 1995. It was acquired by Sun Microsystems in 2008; Sun was in turn acquired by Oracle Corporation in 2010.
- MySQL is developed, distributed, and supported by Oracle Corporation.
- MySQL is named after co-founder Monty Widenius's daughter: My

What is MySQL?

- The data in a MySQL database are stored in tables.
- A table is a collection of related data, and it consists of columns and rows.
- Databases are useful for storing information categorically.
- A company may have a database with the following tables:
  - ✓ Employees
  - ✓ Products
  - ✓ Customers
  - ✓ Orders

MySQL Features

- Fully multi-threaded using kernel threads.
- Works on many different platforms.
- Many column types.
- Very fast joins using an optimized one-sweep multi-join
- Full operator and function support in the SELECT and WHERE parts of queries.
- You can mix tables from different databases in the same query.
- A privilege and password system that is very flexible and secure.
- Handles large databases.
- Tested with a broad range of different compilers.(C/C++)
- No memory leaks.
- Full support for several different character sets.

How MySQL Stores Data

- A MySQL server can store several databases.
- Databases are stored as directories
  - ✓ Default is at /usr/local/mysql/var/
- Tables are stored as files inside each database(directory)
- For each table, it has three files:
  - ✓ table.FRM file containing information about the table structure
  - ✓ table.MYD file containing the row data
  - ✓ table.MYI containing any indexes belonging with this table, as well as some statistics about the table

MySQL Data Types

Text Type	
CHAR( )	A fixed length section from 0 to 255 characters long.
VARCHAR( )	A variable length section from 0 to 255 characters long.
TINYTEXT	A string with a maximum length of 255 characters.
TEXT	A string with a maximum length of 65535 characters.
BLOB	A string with a maximum length of 65535 characters.
MEDIUMTEXT	A string with a maximum length of 16777215 characters.
MEDIUMBLOB	A string with a maximum length of 16777215 characters.
LONGTEXT	A string with a maximum length of 4294967295 characters.
LONGBLOB	A string with a maximum length of 4294967295 characters.

Number Type	
TINYINT( )	-128 to 127 normal - 0 to 255 **UNSIGNED.**
SMALLINT( )	-32768 to 32767 normal - 0 to 65535 **UNSIGNED**.
MEDIUMINT( )	-8388608 to 8388607 normal 0 to 16777215 **UNSIGNED**.
INT( )	-2147483648 to 2147483647 normal 0 to 4294967295 **UNSIGNED**.
BIGINT( )	-9223372036854775808 to 9223372036854775807 normal 0 to 18446744073709551615 **UNSIGNED**.
FLOAT	A small number with a floating decimal point.
DOUBLE(,)	A large number with a floating decimal point.
DECIMAL(,)	A DOUBLE stored as a string , allowing for a fixed decimal point.
TINYINT( )	-128 to 127 normal. - 0 to 255 **UNSIGNED.**

Date Type	
DATE	The is the standard data type to use for storing dates. The date format is YYYY-MM- DD. supported range for dates is 1000-01-01 to 9999-12-31.
DATETIME	DATETIME is similar to DATE, but adds time elements. The standard format is: YYYY-MM-DD HH:MM:SS. Similar to DATE, the range of supported values is 1000-01-01 00:00:00 to 9999-12-31 23:59:59.

Types of Commands

- Data Definition Language (DDL)

  Commands that define a database, including creating, and dropping tables and establishing constraints.

- Data Manipulation Language (DML)

  Commands that send query to a database like add new records, delete records, update the records and select the records.

- Data Control Language (DCL)

  Commands that control a database, including administering privileges and committing data.

Types of Commands

- Data Definition Language (DDL)
  - ✓ Create
  - ✓ Drop
  - ✓ Alter
- Data Manipulation Language (DML)
  - ✓ Select
  - ✓ Update
  - ✓ Insert
  - ✓ Delete
- Data Control Language (DCL)
  - ✓ Grant
  - ✓ Revoke

Basic MySQL Operations

- Create table
- Insert records
- Load data
- Retrieve records
- Update records
- Delete records
- Modify table
- Join table
- Drop table
- Optimize table
- Count, Like, Order by, Group by
- More advanced ones (sub-queries, stored procedures, triggers, views ...)

Basic MySQL Process

- Database process
  - ✓ CREATE DATABASE databaseName;
  - ✓ DROP DATABASE databaseName;
  - ✓ SHOW DATABASES;
  - ✓ USE databaseName;
- DDL process:
  - ✓ SHOW TABLES;
  - ✓ DESCRIBE table;
  - ✓ CREATE TABLE tableName(name1 type1, name2 type2 , ...);
  - ✓ ALTER TABLE tableName add/modify fieldname type,......;
  - ✓ DROP TABLE tableName;
- DML Process:
  - ✓ INSERT INTO TABLE VALUES(value1, value2, ...);
  - ✓ SELECT field1, field2, ... FROM tableName;
  - ✓ UPDATE TABLE tablename SET field = exp/val [where condition];
  - ✓ DELETE FROM tablename [where condition];

PHP MySQL Commands

- mysqli_connect() – Creates a connection
- mysqli_close() – Close the connection
- mysqli_select_db() – select the database
- mysqli_query() - Executes MySQL queries
- mysqli_error() - return the error
- mysqli_errno() - returns the error number of the last MySQL process.
- mysqli_num_rows() - Retrieves the number of rows from a result set.
- mysqli_num_fields() - Retrieves the number of fields from a query

PHP MySQL Commands

- mysqli_affected_rows() – returns the no. of affected rows
- mysqli_fetch_array() - returns a row from a recordset as an associative array / numeric array
- mysqli_fetch_field() - returns an object containing information of a field from a recordset.
- mysqli_fetch_row() - returns a numerical array that corresponds to the fetched row. Returns FALSE if there are no more rows.
- mysqli_list_dbs() — Lists all databases.
- mysqli_free_result() — Free result memory

Create a Connection to a MySQL Database

- Syntax:

```
mysqli_connect(servername, username, password);
```

Parameter	Description
Servername	Optional. Specifies the server to connect
username	Specifies the username to log in with. Default value is the name of the user that owns the server process.
password	Optional. Specifies the password to log in with. Default is “".

MySQL - Create Connection

- Create a connection: Before access data in a database, must create a connection to the database.
- In PHP, this is done with mysql_connect() function.
- mysql_connect(servername,username,password);

```
<?php
$conn = mysqli_connect(“localhost", "root",
     "");
if (!$conn) {
   die("Connection failed:".mysqli_connect_error());
}
echo "Connected successfully";
?>
```

MYSQL – Close

- Close connection: The connection will be closed automatically when the script ends. To close the connection before, use the mysqli_connect() function.
- mysqli_close($conn);

```
<?php
$conn = new mysqli("localhost", "root", "");
if (!$conn) {
    die("Connection failed:" . mysqli_connect_error());
}
echo "Connected successfully";
mysqli_close($conn);
?>
```

My SQL – Select DB

- Database selection can be done by this function

```
mysqli_select_db($dbname);
```

```
<?php
$con=mysqli_connect("localhost","root","");
echo "Connection to the server was successful!";
mysqli_select_db("library") or
      die(mysqli_error());
echo "Database was selected!<br/>";
?>
```

MySQL – Create DB

- All DDL & DML queries can be executed by this function

```
        mysqli_query($query,$con);
```

```
<?php
$con=mysqli_connect("localhost","root","");
if (mysqli_query($con, ”CREATE DATABASE library"))
  echo "Database created";
else
  echo "Error creating DB:" . mysqli_error();
mysqli_close($con);
?>
```

MySQL – DROP DB

```
mysqli_query($query,$con);
```

```
<?php
$con=mysqli_connect("localhost","root","");
if(mysqli_query($con, “DROP DATABASE library"))
  echo "Database Dropped”;
else
  echo "Error in dropping DB:".mysqli_error();
mysqli_close($con);
?>
```

MYSQL - Create Table

```
<?php
$conn = mysqli_connect("localhost", "root", "", "myDB");
if (!$conn) { die("Connection
     failed:".mysqli_connect_error());}

$sql = "CREATE TABLE MyGuests (id INT(6), firstname
     VARCHAR(30),lastname VARCHAR(30), email
     VARCHAR(50), reg_date TIMESTAMP)";

if (mysqli_query($conn, $sql)) {
          echo "Table MyGuests created successfully";
} else {
  echo "Error creating table: " . mysqli_error($conn);
}
mysqli_close($conn);
?>
```

Insert Record into Table

- The INSERT INTO statement is used to add new records to a database table.
- Syntax

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

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

Insert Record into Table

```
<?php
$conn = mysqli_connect("localhost", "root", "", "myDB");
if (!$conn) {
   die("Connection failed: " . mysqli_connect_error()); }

$sql = "INSERT INTO MyGuests (firstname, lastname, email)
VALUES ('John', 'Doe', 'john@example.com')";

if (mysqli_query($conn, $sql)) {
         echo "New record created successfully";
} else {
  echo "Error: " . $sql . "<br>" .mysqli_error($conn);
}
mysqli_close($conn);
?>
```

Insert Record via HTML Forms -1/2

```
//HTML FORM
<html>
<body>
<form action="table.php" method="post">
Name: <input type="text" name="name"/>
ISBN No: <input type="text" name="isbn"/>
Price: <input type="text" name="price" />
<input type="submit" />
<input type="reset" />
</form>
</body>
</html>
```

Insert Record via HTML Forms -2/2

```
//PHP Content
<?php
$con = mysqli_connect("localhost", "root", "");
mysqli_select_db("library");

$n=$_POST['name'];  //Retriving Data from HTML
$i=$_POST['isbn'];   //Retriving Data from HTML
$p=$_POST['price'];   //Retriving Data from HTML
$query = "insert into book values('$n','$i','$p')";

$res = mysqli_query($query);
if ($res == true) {
   echo "Insertion Success ...";}
else {
   echo "Error .... ";  }
 mysqli_close($con);
 ?>
```

Select Data From a Database Table

- Syntax

  SELECT column_name(s) FROM table_name

- mysqli_fetch_array() function to return the first row from the recordset as an array.
- Each call to mysqli_fetch_array() returns the next row in the recordset. The while loop loops through all the records in the recordset.

Select from Table

```
<?php
$con = mysqli_connect(“localhost","root","");
mysqli_select_db("myDB");
$query = "select * from MyGuests";
$res = mysqli_query($query);
mysqli_close($con);
?>
```

The WHERE clause

- The WHERE clause is used to extract only those records that fulfil a specified criterion.
- Syntax

```
SELECT column_name(s)
FROM table_name
WHERE column_name operator value
```

Select from Table - Where

```
<?php
$con = mysqli_connect("localhost", "root", "");
mysqli_select_db("myDB");
$query = "select * from MyGuests where id=‘1002'";
$res = mysqli_query($query);
mysqli_close($con);
?>
```

The ORDER BY Keyword

- The ORDER BY keyword is used to sort the data in a recordset.
- The ORDER BY keyword sort the records in ascending order by default.
- If you want to sort the records in a descending order, you can use the DESC keyword.
- Syntax

```
SELECT column_name(s)
FROM table_name
ORDER BY column_name(s) ASC|DESC
```

Select from Table - Orderby

Order by Ascending & Descending Order

```
<?php
$con = mysqli_connect("localhost", "root", "");
mysqli_select_db("library");
$query1 = "select * from book order by price";
$query2 = "select * from book order by price desc";
$res1 = mysqli_query($query1);
$res2 = mysqli_query($query2);
mysqli_close($con);
?>
```

Update Data In a Database

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

```
UPDATE table_name
SET column1=value, column2=value2,…
WHERE some_column=some_value
```

Update Table

```
<?php
$con = mysqli_connect("localhost", "root", "");
mysqli_select_db("library");
$query = "update book set name='phpmysql' where
    name='php' AND price='350'";
$res = mysqli_query($query);
mysqli_close($con);
?>
```

Delete Data In a Database

- The DELETE FROM statement is used to delete records from a database table.
- Syntax

```
DELETE FROM table_name
WHERE some_column = some_value
```

Delete Record

```
<?php
$con = mysqli_connect("localhost", "root", "");
mysqli_select_db("library");
$query = "delete from book where price='350'";
$res = mysqli_query($query);
mysqli_close($con);
?>
```

Drop Column

```php
<?php
$con = mysqli_connect("localhost", "root", "");
mysqli_select_db("library");
$query = "alter table book drop price";
$res = mysqli_query($query);
mysqli_close($con);
?>
```

Alter Table

```php
<?php
$con = mysqli_connect("localhost", "root", "");
mysqli_select_db("library");
$query = "alter table book add column author
    varchar(25)";
$query1 = "alter table book change isbn isbnno
    varchar(25)";
$res = mysqli_query($query);
mysqli_close($con);
?>
```

My SQL – Error & Error No

- Mysqli_error() – describes the error for the last MySQL process.
- mysqli_errno() – returns the error number for the last process.

My SQL – Error & Errno

```
<?php
$con = mysqli_connect("localhost","root","");
$db = mysqli_select_db("wrongdb",$con);
If (!$db)
{
        echo "error – ".mysqli_error();
        echo "Error number – ".mysqli_errno();
}
mysqli_close($con);
?>
```

Num rows

- Mysqli_num_rows()
  - ✓ Retrieves the number of rows from a result set. This command is only valid for statements like SELECT or SHOW that return an actual result set.

Num rows

```
<?php
$con = mysqli_connect("localhost", "root", "");
mysqli_select_db("library");
$query = "select * from book";
$num_rows = mysqli_num_rows($query);
echo "$num_rows Rows were selected";
mysqli_close($con);
?>
```

Num Fields

- mysqli_num_fields() – Retrieves the number of fields from a result set.

Num Fields

```
<?php
$con=mysqli_connect("localhost","root","");
mysqli_select_db("library");
$query = "select * from book";
$num_fields = mysqli_num_fields($query);
echo "$num_fields Fields were selected";
mysqli_close($con);
?>
```

Affected Rows

- Mysqli_affected_rows() – The number of rows in a result set on success or FALSE on failure. It is valid for insert, delete, and update commands only.

```
mysqli_affected_rows($recordset);
```

Affected Rows

```
<?php
$con = mysqli_connect(“localhost","root","") ;
mysqli_select_db("library");
$query = "update book set name='phpmysql'
    where name='php' AND price='350'";
$res = mysqli_query($query);
$num_rows = mysqli_affected_rows($res);
echo "$num_rows Rows were Affected";
mysqli_close($con);
?>
```

Fetch Rows

- mysqli_fetch_row() - returns a numerical array that corresponds to the fetched row. Returns FALSE if there are no more rows.

```
mysqli_fetch_row($recordset);
```

Fetch Rows

```
<?php
$con = mysqli_connect("localhost","root","");
mysqli_select_db("library");
$result = mysqli_query("select * from book");
while ($r1 =mysqli_fetch_row($result))
{
   echo $r1[0]. $r1[1].$r1[2]."<br>";
        // table has 3 fields only
}
mysqli_close($con);
?>
```

Fetch Array

- Extended version of mysqli_fetch_row.
- Returns a numeric/associative array that corresponds to the fetched row and moves the internal data pointer ahead or FALSE if there are no more rows.

```
mysqli_fetch_array($recordset);
```

Fetch Array - Method 1

```php
<?php
$con = mysqli_connect("localhost","root","");
mysqli_select_db("library");
$result = mysqli_query("select * from book");
while($row = mysqli_fetch_array($result))
{
 print_r($row)
}
mysqli_close($con);
?>
```

Fetch Array - Method 2

```
<?php
$con = mysqli_connect("localhost","root","");
mysqli_select_db("library");
$res = mysqli_query("select * from book");
while($row = mysqli_fetch_array($res))
{
    echo $row['name'], " ";
    echo $row['isbn'], " ";
    echo $row['price'], " ";
}
mysqli_close($con);
?>
```

Fetch Associative

- Returns an associative array that corresponds to the fetched row and moves the internal data pointer ahead or FALSE if there are no more rows. Here, field names are array keys .

```
mysqli_fetch_assoc($recordset);
```

Fetch Associative

```
<?php
$con = mysqli_connect("localhost","root","");
mysqli_select_db("library");
$result = mysqli_query("select * from book");
while($row = mysqli_fetch_assoc($result))
{
   print_r($row)
}
mysqli_close($con);
?>
```

Fetch Field

- Returns an object containing field information. This function can be used to obtain information about fields in the provided query result.

```
mysqli_fetch_field($recordset);
```

Fetch Field

- The properties of the object are:
  - ✓ name - column name
  - ✓ table - name of the table the column belongs to
  - ✓ max_length - maximum length of the column
  - ✓ not_null - 1 if the column cannot be NULL
  - ✓ primary_key - 1 if the column is a primary key
  - ✓ unique_key - 1 if the column is a unique key
  - ✓ multiple_key - 1 if the column is a non-unique key
  - ✓ numeric - 1 if the column is numeric
  - ✓ blob - 1 if the column is a BLOB
  - ✓ type - the type of the column
  - ✓ unsigned - 1 if the column is unsigned
  - ✓ zerofill - 1 if the column is zero-filled

Fetch Field

```php
<?php
$con = mysqli_connect("localhost", "root", "");
$db_selected = mysqli_select_db("library",$con);
$result = mysqli_query("SELECT * from book",$con);
while ($property = mysqli_fetch_field($result))
{
   echo "Field name: " . $property->name . "<br />";
   echo "Table name: " . $property->table . "<br />";
   echo "Default value: " . $property->def . "<br />";
   echo "Max length: " . $property->max_length . "<br />";
   echo "Not NULL: " . $property->not_null . "<br />";
   echo "Primary Key: " . $property->primary_key . "<br />";
   echo "Unique Key: " . $property->unique_key . "<br />";
   echo "Mutliple Key: " . $property->multiple_key . "<br />";
   echo "Numeric Field: " . $property->numeric . "<br />";
   echo "BLOB: " . $property->blob . "<br />";
   echo "Field Type: " . $property->type . "<br />";
   echo "Unsigned: " . $property->unsigned . "<br />";
   echo "Zero-filled: " . $property->zerofill . "<br /><br />";
}
mysqli_close($con);
?>
```

Between Function

- The MySQL BETWEEN Condition is used to retrieve values within a range in a SELECT, INSERT, UPDATE, or DELETE statement.
- Syntax

```
expression BETWEEN value1 AND value2;
```

Between Function

```
<?php
$con = mysqli_connect("localhost","root","");
mysqli_select_db("library");
$result = mysqli_query("select * from book");
$query = "select price from book where price
    BETWEEN 300 AND 325";
$res = mysqli_query($query);
mysqli_close($con);
?>
```

NOT Between Function

- MySQL NOT BETWEEN AND operator checks whether a value is not present between a starting and a closing expression
- Syntax:

```
expr NOT BETWEEN min AND max
```

NOT Between Function

```
<?php
$con = mysqli_connect("localhost","root","");
mysqli_select_db("library");
$result = mysqli_query("select * from book");
$query = "select price from book where price
    NOT BETWEEN 300 AND 325";
$res = mysqli_query($query);
mysqli_close($con);
?>
```

Like Function

- MySQL LIKE operator along with WILDCARDS finds a string of a specified pattern within another string.
- In a more technical note, LIKE operator does pattern matching using simple regular expression comparison.

Wildcards	Description
%	Matches any number of characters including zero.
_ (underscore)	Matches exactly one character.

Example - Using % wildcard

- The % wildcard works in the MySQL LIKE condition. We want to find all of the customers whose last_name begins with ‘Sm’.

```
SELECT customer_name
FROM customers
WHERE last_name LIKE ‘Sm%';
```

- You can also using the % wildcard multiple times within the same string. F

```
SELECT customer_name
FROM customers
WHERE last_name LIKE ‘%it%’;
```

Example - Using _ wildcard

- The _ wildcard (underscore) works in the MySQL LIKE condition. Remember that _ wildcard is looking for only one character.

```
SELECT supplier_name
FROM suppliers
WHERE supplier_name LIKE ‘Sm_th';

SELECT *
FROM suppliers
WHERE account_number LIKE '12345_';
```

Like Function

```
<?php
$con = mysqli_connect("localhost","root","");
mysqli_select_db("library");
$result = mysqli_query("select * from book");
//Name Starting with A
$query =mysqli_query("select * from book
     where name like 'a%'");
//Name Ending with N
$query1 =mysqli_query("select * from book
     where name like '%n'";
mysqli_close($con);
?>
```

LIKE NOT Operator

- The % wilcard with the NOT Operator. You could also use the MySQL LIKE condition to find suppliers whose name does not start with 'G'.

```
SELECT supplier_name
FROM suppliers
WHERE supplier_name NOT LIKE ‘G%';
```

Truncate

- The TRUNCATE TABLE command deletes the data inside a table, but not the table itself.
  - ✓ TRUNCATE TABLE table_name;

Truncate

```
//Deletes only Records not Fields
<?php
$con = mysqli_connect("localhost","root","");
mysqli_select_db("library");
$query = "truncate table book";
$res = mysqli_query($query);
mysqli_close($con);
?>
```

MySQL Client info

- mysqli_get_client_info() returns a string that represents the client library version.

```
mysqli_get_client_info();
```

```
<?php
echo "MySQL client info: " .
    mysqli_get_client_info();
?>
```

MySQL free result

- mysqli_free_result() will free all memory associated with the result identifier result.
- mysqli_free_result() only needs to be called if you are concerned about how much memory is being used for queries that return large result sets. All associated result memory is automatically freed at the end of the script's execution.

MySQL free result

- Clears the record set. mysqli_free_result($recordset);

```
<?php
$con = mysqli_connect("localhost","root","");
$db_selected = mysqli_select_db("library",$con);
$result = mysqli_query("SELECT * from book",$con);
print_r($row=mysqli_fetch_row($result));
mysqli_free_result($result);
print_r($row=mysqli_fetch_row($result)); //Shows Error
mysqli_close($con);
?>
```

List the Fields with properties

This program lists all databases

```
<?php
$con = mysqli_connect("localhost", "root", "root");
if (!$con) {
    die(mysqli_connect_error());
}
$db_list = mysqli_query($con, "SHOW DATABASES");
while ($db = mysqli_fetch_object($db_list)) {
    echo $db->Database . "<br />";
}
mysqli_close($con);
?>
```

Multiple Database

- Multiple databases can be used at a time.

```
<?php
$con = mysqli_connect(“localhost","root","");

$db1 = mysqli_select_db("library"); //DB-1
$db2 = mysqli_select_db("lib"); //DB-2

$result1 = mysqli_query("select * from book", $db1);
$result2 = mysqli_query("select * from libbook", $db2);
mysqli_close($con);
?>
```

Thank You !!!

😎