A·P·U
ASIA PACIFIC UNIVERSITY
OF TECHNOLOGY & INNOVATION

AAPP009-4-2 Web Development

Web Database using MySQL

Topic & Structure of The Lesson

- Storage
  - Temporary and Permanent
  - Unstructured and Structured
- Ways to structure information
- Simple databases
- Relationships, example relationship
- Some Database Terminology
- SQL – Structured Query Language

Learning Outcomes

- **At the end of this topic, You should be able to**
  - Understand relational database
  - Understand how to write SQL statements for CRUD
  - Able to manipulate form inputs into database

Web and Databases: An Essential Relationship

- **Data Storage:** Databases provide a structured way to store, retrieve, and manage data. They are essential for websites that need to store user information, content, or any other data that the website needs to function.
- **Dynamic Content:** Databases allow websites to display dynamic content. This means that the content of the website can change in response to different contexts or conditions.
- **User Interaction:** Databases enable user interactions on websites. For example, when a user posts a comment on a blog, that comment is stored in a database and then retrieved and displayed each time the blog post is viewed.
- **Data Security:** Secure database design is crucial for protecting sensitive information. This includes encrypting data, setting up proper access controls, and regularly backing up data.
- **Scalability:** As a website grows, its database needs to handle increasing amounts of data. Databases can be scaled up or out to meet these demands, ensuring that the website continues to perform well as its data needs increase.

Designing a Database: Structuring Information

- **Repetitive Form:** Data often follows a repetitive form, meaning the type of data remains consistent across records.
- **Repetitive Content:** The content of data can also be repetitive. For example, in a database of people, you might have multiple records where the age is 30.
- **Data Structure:** Consider a dataset of 15 people where each person has an age, name, and height. This gives us 15 records, each with 3 fields. The format for each record is:

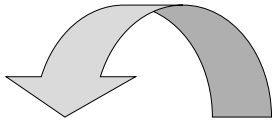
[whole number (age)], [text (name)], [fractional number (height)].

- **Tabular Format:** This structured data can be efficiently stored in a table format, where each row represents a record (a person in this case), and each column represents a field (age, name, height).
- **Database Design:** When designing a database, it's important to identify these patterns and structure your database accordingly. This allows for efficient storage and retrieval of data.

Table (with repeated information)

No.	First Name	Last Name	Origin
1	Danny	DeVito	United States of America
2	Arnold	Schwarzenegger	Austria
3	Halle	Berry	United States of America
4	Patrick	Stewart	United Kingdom
5	Karisma	Kapoor	India
6	Hugh	Jackman	Australia
7	Alec	Guinness	United Kingdom
8	Jack	Nicholson	United States of America
9	Kylie	Minogue	Australia

Table (no repeated information)



No.	First Name	Last Name	Origin
1	Danny	DeVito	1
2	Arnold	Schwarzenegger	2
3	Halle	Berry	1
4	Patrick	Stewart	3
5	Karisma	Kapoor	5
6	Hugh	Jackman	4
7	Alec	Guinness	3
8	Jack	Nicholson	1
9	Kylie	Minogue	4

No.	Country Name
1	United States of America
2	Austria
3	United Kingdom
4	Australia
5	India

This is an example of a simple one-to-many *relationship*.

Some Database Terminology

- Rows and Columns
  - Broad view of the table – each row is a record, each column is a set of fields
  - This is analogous to a spreadsheet view
- Records and Fields
- Detailed view of the information
- Each record (entry) in the database is made up of fields, possibly with some fields from joined tables

Understanding Relationships and Joins in RDBMS

- **Joining Tables:** The ability to join tables is a powerful feature of RDBMS, allowing for complex data manipulation and retrieval. However, it can also introduce complexity.
- **Relational Databases:** Database systems that support join operations are known as Relational Database Management Systems (RDBMS). They provide a structured way to store, organize, and retrieve information.
- **Tools for Managing Joins:** To construct, utilize, and maintain joins between tables, it's often most efficient to use the tools built into RDBMS software. Examples of such software include MS Access, MySQL, MS SQL, OracleDB, and others.

SQL – Structured Query Language

- Structured Query Language (SQL), is a set of commands that all programs and users may use to access data within databases
- Application programs and tools often allow users to access SQL databases without directly using SQL, but these applications in turn must use SQL when executing the user's request

Basic Statements (manage Database and Table)

SQL Statement	Description
Create database	Creates a new database.
Drop database	Deletes an existing database.
Create table	Creates a new table in the database.
Modify table	Alters the structure of an existing table, such as adding or deleting columns.
Drop table	Deletes an existing table from the database.

Basic Statements (Manage Record)

SQL Statement	Description
Insert records	Adds new records (rows) to a table.
Retrieve records	Fetches records from a table based on specified conditions.
Update records	Modifies existing records in a table.
Delete records	Removes existing records from a table.
Join table	Combines rows from two or more tables based on related columns between them.
Count, Like, Order by, Group by	These are SQL clauses: 'Count' calculates the number of rows that match a specified condition; 'Like' is used in a WHERE clause to search for a specified pattern in a column; 'Order by' sorts the result-set in ascending or descending order; 'Group by' groups rows that have the same values in specified columns into aggregated data (like sum, average, or count).

Insert Record

- INSERT INTO table_name VALUES (value1, value2,....)

- Example

  INSERT INTO student VALUES (101, 'Shannon', 'BCB', 'A');

Student_ID	Name	Major	Grade
101	Shannon	BCB	A

INSERT INTO student(Student_ID, Name) VALUES (102, 'Mike');

Student_ID	Name	Major	Grade
101	Shannon	BCB	A
102	Mike		

Retrieve Record

- SELECT what_columns FROM table or tables WHERE condition
- Example

SELECT major, grade FROM student WHERE name='Shannon';

Major	Grade
BCB	A

SELECT * FROM student;

Student_ID	Name	Major	Grade
101	Shannon	BCB	A
102	Mike	BBMB	A
103	Wang	MCDB	A

Update Record

- UPDATE table_name SET which columns to change WHERE condition

- Example

  UPDATE student SET grade='B' WHERE name='Shannon';

Student_ID	Name	Major	Grade
101	Shannon	BCB	B

Delete Record

- DELETE FROM table_name WHERE condition

- Example
  DELETE FROM student WHERE name='Shannon';

- MUST include where condition, otherwise all records will be deleted.
  DELETE FROM student;

MySQL Connectivity

- **`mysqli_connect()`**
  - The **`mysqli_connect()`** function opens a non-persistent MySQL connection.
  - This function returns the connection on success, or FALSE and an error on failure.
- Syntax
  - **`mysqli_connect(server,user,pwd,dbname)`**

- **`mysqli_close()`**
  - The **`mysqli_close()`** function closes a non-persistent MySQL connection.
  - This function returns TRUE on success, or FALSE on failure.
- Syntax:
  - **`mysqli_close(connection)`**

MySQL Connectivity

- **`mysqli_query()`**
  - The **`mysqli_query()`** function executes a query on a MySQL database.
  - This function returns the query handle for SELECT queries,, or FALSE on failed queries.
- Syntax
  - **`mysqli_query(query,connection)`**

- **`mysqli_fetch_array()`**
  - The **`mysqli_fetch_array()`** function returns a row from a recordset as an array on success or FALSE on failure or when there are no more rows.
- Syntax
  - **`mysqli_fetch_array(data,array_type)`**

Using PHP and MySQL to Insert Data into a Database

1. User Input (Web Form)
2. Connect to Database (PHP mysqli_connect)
3. Retrieve Data (PHP $_POST)
4. Create SQL Query (PHP String with SQL syntax)
5. Execute SQL Query (PHP mysqli_query)
6. Close Database Connection (PHP mysqli_close)

Creating a User Input Form with HTML

```
<form action="insert.php" method="post">

    Name:<input type="text" name="name">

    Email <input type="email" name="email" >

    Comments :<input type="text" name="comment">

    <input type="submit" name="submit" value="Proceed">

</form>
```

This line creates a form that, when submitted, sends the form data to a file named “insert.php”. The method="post" attribute specifies that the form data should be sent as a POST request, which means the data is included in the body of the HTTP request and not visible in the URL.

Establishing a MySQL Database Connection with PHP

This line is attempting to open a connection to a MySQL database server. The mysqli_connect() function takes four parameters: the hostname of the database server (in this case, “localhost”), the username to log in as (here, “root”), the password for that user (also “root”), and the name of the database you want to connect to (here, “db”).

```php
$con=mysqli_connect("localhost","root","root","db");

// Check connection
if(mysqli_connect_errno())
{
echo "Failed to connect to MySQL: ".mysqli_connect_error();
}
```

This part of the code checks if the connection was successful. If mysqli_connect_errno() returns a non-zero value, it means there was an error in the connection. The code then prints an error message that includes the specific error returned by mysqli_connect_error()

Inserting User Data into a Database with PHP and MySQL

```php
// Receives values from Form, assigns values entered to table users
$nm = $_POST["name"];
$em = $_POST["email"];
$cmt = $_POST["comment"];
```

These lines are retrieving data that has been posted from a form on a webpage. The $_POST superglobal array is used to collect data from a form using the POST method in PHP. The array indices "name", "email", and "comment" correspond to the name attributes of the form inputs on the HTML page.

```php
// Declares SQL statement that will add data to the database
$sql = "INSERT INTO users (usrName, usrEmail, usrComments) VALUES ('$nm', '$em', '$cmt')";
```

This line is creating an SQL query that will insert the retrieved data into a table named users. The usrName, usrEmail, and usrComments are column names in the users table, and $nm, $em, and $cmt are the corresponding values to be inserted.

```php
// Just runs the SQL query (but better if we checked for errors)
if(!mysqli_query($con,$sql))
{
die('Error: '.mysqli_error($con));
}
```

This part of the code executes the SQL query using the mysqli_query() function. If there's an error in executing the query, it will stop the script and print an error message.

```php
mysqli_close($con);
```

This line closes the connection to the MySQL database.

Summary of Main Teaching Points

- Storage
- Ways to structure information
- Simple databases
- Relationships, example relationship
- Some Database Terminology
- SQL – Structured Query Language

Q & A