TAREA 02. TRABAJAR CON BASES DE DATOS EN PHP

DESARROLLO WEB ENTORNO SERVIDOR

Nombre: María Palomares Gallo

Asignatura: Desarrollo Web en entorno Servidor.

Ciclo: Desarrollo Aplicaciones Web

1. **No pienses en una aplicación web ni en una aplicación de escritorio concreta y cita al menos tres tipos de sistemas que podrían utilizarse para organizar la información del almacén. Estos tipos de sistemas no tienen porqué ser necesariamente una base de datos relacional (existen otras formas de almacenar la información en un ordenador), aunque una base de datos relacional es uno de los tipos de sistemas de almacenamiento de la información que deberías contemplar. En este apartado no se espera que indiques bases de datos concretas (como MySQL, MongoDB, etc.) sino tipos de sistemas de almacenamiento en general, aunque puedes proporcionar un ejemplo de cada tipo (por ejemplo, MySQL es un tipo de base de datos relacional).**

   Base de datos relacional: Los datos se almacenan en tablas que se dividen en filas y columnas, siguiendo un modelo relacional. Cada tabla puede estar conectada con otra por medio de claves primarias o foráneas. Este tipo de sistema es muy util ya que se produce una organización de los datos estructurada y poder realizar consultas desde base de datos concretas como es MySQL.

   Base de datos noSQL: Este tipo de bases de datos se utiliza cuando los datos que queremos introducir no encajan bien con el modelo tabular de las bases de datos relacionales. Podemos ver diferentes tipos como bases de datos de documentos, de clave-valor, bases de datos en grafos y bases de datos de columnas anchas. Un ejemplo claro de este tipo de base de datos es MongoDB.

   Almacenamiento en la nube: Utilizamos los servicios de la nube para almacenar y gestionar datos. El lado positivo de este tipo de almacenamiento es que puede ser escalable y se puede acceder desde cualquier lugar simplemente disponiendo de conexión a internet.

2. **Partiendo del ejercicio anterior, reflexiona sobre cuáles son los tipos sistemas más convenientes de cara a desarrollar una aplicación web desde PHP.**

   Los sistemas que veo más convenientes partiendo del ejercicio anterior es la base de datos relacional, ya que es la que nos permite gestionar datos estructurados y transaccionales con gran facilidad. También php cuenta con bibliotecas y extensiones que permiten conectar y gestionar las bases de datos de forma mucho más fácil. Estas son las más indicadas para trabajar con aplicaciones donde almacenemos productos, nombres, etc.

   Las bases de datos noSQL también pueden ser muy útiles para esto, porque nos permiten almacenar tanto datos estructurados como no estructurados y así podemos tener campos flexibles o manejar inventarios con estructuras de productos muy variadas. Este también puede conectarse fácilmente con PHP por medio de extensiones. Este tipo de base de datos es muy útil si se requiere alta escalabilidad y rendimiento, especialmente en aplicaciones con grandes volúmenes de datos.

La nube por último es muy buena opción si se trata de aplicaciones con un alto volumen de datos con alta escalabilidad y flexibilidad. PHP puede interactuar con servicios de almacenamiento en la nube mediante SDKs. Es ideal para almacenar datos de respaldo, archivos grandes (como imágenes de productos o documentos) o integrar servicios como análisis en tiempo real.

3. **Si usamos una base de datos MySQL o MariaDB, ¿con qué tipo de sistema de almacenamiento de información de los citados por ti en el ejercicio anterior encajaría?**

   Lo metería en base de datos relacional, ya que MySQL trabaja por medio de la inserción de datos por medio de tablas que se dividen en filas y columnas que se pueden comunicar entre ellas por claves primarias y foráneas. También podemos hacer consultas complejas por medio de esta base de datos para la consulta, modificación o eliminación de datos.

4. **Desde PHP se puede acceder a una base de datos MySQL o MariaDB a través de diferentes formas. Cita al menos dos formas diferentes y explica sus diferencias.**

   Podemos hacerlo de dos maneras, una es MySQLi y PDO. Las diferencias que encontramos es que MySQLi solamente tiene compatibilidad con MySQL y MariaDB, mientras que PDO es una forma que tiene más compatibilidad con otras bases de datos. Otra diferencia es que MySQLi tiene una interfaz orientada a objetos y procedural, mientras PDO solamente está orientada a objetos. Como última diferencia podemos decir que MySQLi está orientada más a personas con menos conocimientos, mientras que usar PDO requiere un conocimiento más avanzado sobre las bases de datos.

5. **Investiga qué diferencias hay entre MySQL y Apache Cassandra, y reflexiona sobre qué limitaciones o ventajas tendría el uso de una u otra de cara a desarrollar la aplicación web para esta empresa.**

   Modelo de base de datos: La principal diferencia que hay entre MySQL y Cassandra es que son diferentes tipos de bases de datos, siendo la primera una base de datos relacional que se basa en tablas formadas por filas y columnas, mientras que Cassandra es una base de datos noSQL basada en familias de columnas, esto quiere decir que trabaja también con datos no estructurados y semi estructurados.

   Escalabilidad: Mientras que MySQL escala de forma vertical, aumentando la capacidad del servidor, esto hace que sea difícil trabajar con aplicaciones que tengan datos masivos o gran cantidad de datos. Cassandra escala de forma horizontal, permitiendo añadir nuevos nodos, esto hace que sea más fácil trabajar con gran cantidad de tráfico de datos.

   Consistencia y disponibilidad: Por un lado, MySQL garantiza consistencia fuerte mediante las propiedades ACID, lo que es crucial para aplicaciones que requieren integridad transaccional. Mientras que Cassandra, sigue un modelo de consistencia

eventual, priorizando la disponibilidad y tolerancia a particiones, lo que puede ser problemático para datos que requieren sincronización instantánea.

Rendimiento: MySQL es adecuado para realizar consultas complejas con una cantidad de datos moderados, pero puede presentar problemas en escenarios de alta concurrencia o escrituras demasiado complejas e intensivas. Por otro lado, Cassandra está optimizado para cargas de trabajo pesadas de escritura y entornos distribuidos, como análisis en tiempo real.

La conclusión que saco después de recabar las diferencias más notorias entre estas dos, es que según el tipo de aplicación que queramos elaborar debemos usar un tipo u otro. Si queremos una aplicación con consistencia y destinada más a la transacción sería más indicado trabajar con MySQL, mientras que si en nuestra aplicación se requiere escalabilidad masiva y manejo de datos distribuidos con alta disponibilidad, sería más indicado trabajar con Cassandra.