educación IT
CODE YOUR FUTURE

Programación Web .NET Core

Módulo 4

educación IT
CODE YOUR FUTURE

Planificación del diseño de la base de datos

Planificación del diseño de la base de datos

Una vez que se conocen bien los requisitos funcionales y técnicos, comienza el diseño de la implementación física de la aplicación.

Los objetos físicos más importantes para planificar son las bases de datos.

Aunque no todas las aplicaciones web utilizan bases de datos para el almacenamiento de la información, son un objeto subyacente para la mayoría de los sitios y se utilizan en casi todos los proyectos.

Modelado lógico

Se puede comenzar el diseño de datos a un alto nivel con la creación de **diagramas de modelo de dominio** en UML y **diagramas de modelo de datos lógicos** (LDM).

Un **diagrama de modelo de dominio**, también conocido como ***modelo de datos conceptual***, muestra la estructura conceptual de alto nivel.

> Por ejemplo, en una aplicación web de comercio electrónico, incluye los conceptos de *clientes*, *carritos de compras* y *productos*. Este modelo no incluye detalles de las propiedades de cada concepto, pero muestra las **relaciones** entre ellos y sirve para registrar las conversaciones iniciales con las partes interesadas.

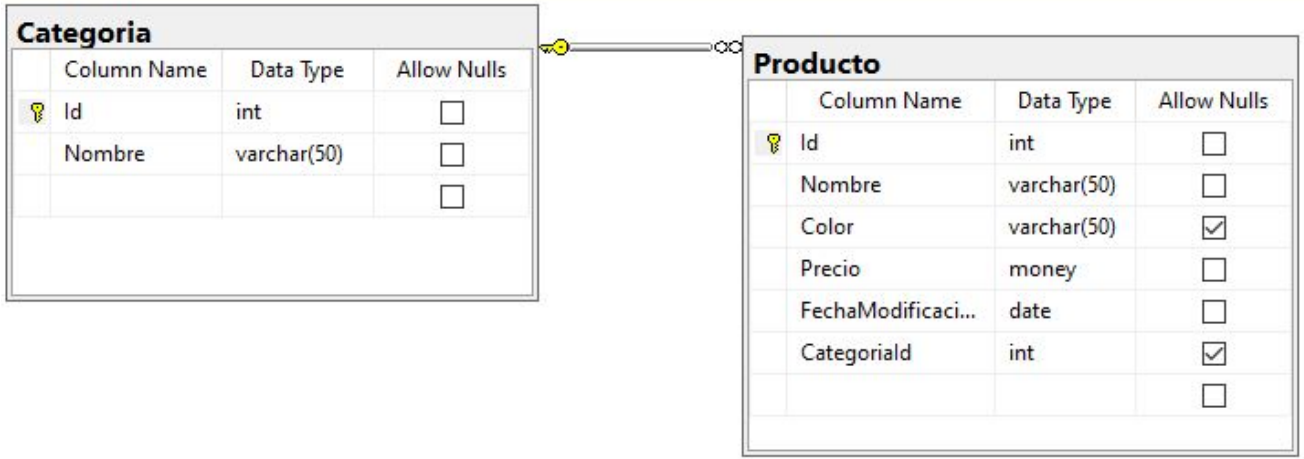
En esencia, un LDM es un diagrama de modelo de dominio con detalles adicionales agregados. Se pueden usar diagramas LDM para completar más detalles, como propiedades y tipos de datos, para los conceptos que se definieron o en el modelo de dominio.

Estructura física de la base de datos

En un plan de proyecto, se deben considerar los siguientes objetos de base de datos:

- **Tablas:** Al definir una tabla, cada columna debe tener el tipo de dato adecuado: un entero, cadena, moneda, fecha y otros. Una clave principal debe identificar de **forma única** cada registro. Esto es esencial para definir las relaciones con registros en otras tablas.
- **Procedimientos almacenados:** Son objetos que almacenan consultas complejas y de modificación con control de las transacciones.
- **Seguridad.** Se debe considerar cómo la aplicación web se autenticará con el servidor de la base de datos y cómo autorizará el acceso a cada tabla de la base de datos.

Un modelo de datos físicos es un diagrama que representa tablas, columnas, tipos de datos y relaciones entre tablas. Veamos un esquema de ejemplo en la pantalla siguiente.


Categoria
Column Name
Data Type
Allow Nulls
Id
int
Nombre
varchar(50)
Producto
Column Name
Data Type
Allow Nulls
Id
int
Nombre
varchar(50)
Color
varchar(50)
Precio
money
FechaModificaci...
date
CategoriaId
int

Trabajar con los DBA (administradores de bases de datos)

A veces, el equipo de desarrollo tiene control total sobre la base de datos que subyace a la aplicación web. Esto sucede, por ejemplo, cuando la organización es pequeña o cuando la aplicación web tiene un servidor de base de datos sin datos críticos para la empresa.

Sin embargo, en organizaciones más grandes, puede haber un equipo dedicado de **administradores de bases de datos (DBA)** cuyo trabajo es garantizar la integridad de los datos según la política de almacenamiento de la organización.

Si el equipo de DBA administra la base de datos del proyecto, es esencial la comunicación con ellos para conocer sus requisitos. Con frecuencia imponen una lista de requisitos técnicos, los cuales otras partes interesadas podrían no entender. Los administradores de bases de datos son responsables de crear bases de datos en los servidores o clústeres adecuados y de asignar permisos. **Los DBA son contribuyentes críticos en la entrega de la aplicación web**.

Diseño de bases de datos en desarrollo Ágil y programación extrema

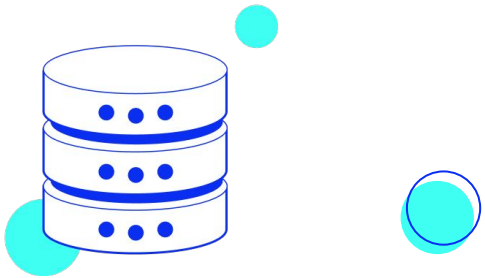
El desarrollo ágil y la programación extrema se caracterizan por una cantidad relativamente pequeña de planificación y documentación, y reconocer que **es probable que los requisitos cambien a lo largo del proyecto de desarrollo.**

Durante la fase de planificación inicial del proyecto, **solo se crea el modelo de dominio**. No se desarrollarán el LDM ni modelos de datos físicos hasta que se escriba el código que implemente los requisitos funcionales.

Durante la **fase de desarrollo,** justo antes de escribir el código, **se discutirán los requisitos con los usuarios y DBA**, y se creará el **LDM y modelos de datos físicos.**

En desarrollo ágil y programación extrema, el diseño de la base de datos cambia a lo largo del proyecto hasta el despliegue. Por lo tanto, los desarrolladores deberían poder modificarla siempre que sea necesario, sin consultar a los DBA o tener que cumplir con políticas de datos complejas. Por esta razón, **la base de datos debe estar alojada en un servidor de desarrollo dedicado.**

Si la base de datos ya existe, se deberá contar con **una copia en un servidor de desarrollo** para garantizar que el código de desarrollo no modifique por error los datos de producción.



Planificación para despliegue distribuido

Planificación de aplicaciones distribuidas

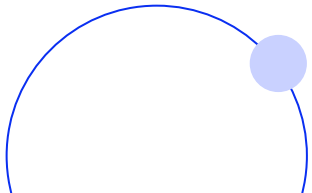
Capas

- Presentación.
- Lógica de negocios.
- Acceso a los datos.
- Base de datos.

Comunicación.

Seguridad.

Para una aplicación web pequeña, con poco tráfico de usuarios, se puede optar por alojar todos los componentes en un solo servidor. Aunque, a medida que la aplicación crece, muchas veces se recurre a un ***despliegue distribuido***, en el que **diferentes servidores alojan componentes separados de la aplicación**.



Las *aplicaciones web distribuidas* utilizan una arquitectura en capas:

- **Capa de presentación:** Los componentes de esta capa implementan la interfaz de usuario y lógica de presentación. Si se está construyendo una *Aplicación web MVC*, ***Vistas*** es la capa de presentación.
- **Capa de lógica empresarial:** Los componentes de esta capa implementan objetos comerciales de alto nivel, como productos, o clientes. En MVC, los controladores forman la capa de negocio.

- **Capa de acceso a datos:** Los componentes de esta capa implementan operaciones de acceso a la base de datos y a los objetos de base de datos, como tablas y procedimientos almacenados. Por ejemplo, un objeto comercial de producto podría incluir datos de las tablas de la base de datos ***Producto*** y ***Stock***. En MVC, los modelos representan a la capa de acceso a datos.

- **Capa de base de datos:** La propia base de datos es una capa en su aplicación web. Puede alojar cada capa en diferentes servidores. Por ejemplo, la capa de presentación se aloja en un servidor IIS, la lógica empresarial y las capas de acceso a datos están alojadas en servidores dedicados de nivel medio, y la base de datos está alojada en un SQL Server dedicado, o en Microsoft Azure.

Comunicación entre capas

Cuando un solo servidor aloja todos los componentes de una aplicación web, la presentación, la lógica empresarial y los componentes de acceso a datos se ejecutan en un solo proceso en la memoria del servidor web.

Cuando la aplicación se ejecuta en **diferentes capas en diferentes servidores**, se considerará:

1. ***¿Cómo intercambia información y mensajes a cada capa?***
2. ***¿Cómo cada servidor autentica y asegura las comunicaciones con otros servidores?***

Veamos...

1. La comunicación de información y seguridad se hace de diferentes formas entre las distintas capas: En el navegador web se ejecuta la capa de presentación, se comunica con el servidor vía la transferencia de hipertexto Protocolo (**HTTP**). Suele requerir alguna técnica de autenticación, entre el servidor web y el servidor de nivel medio. El mecanismo de comunicación y seguridad usado para la comunicación depende de la tecnología que se utiliza para crear los componentes de negocio.

Dos tecnologías comunes son las siguientes: **Web API** y **Windows Communication Foundation (WCF)**.

2. El servidor de nivel medio envía consultas T-SQL al servidor de base de datos, que se autentica en la base de datos con las credenciales necesarias. Estas a menudo se incluyen en la cadena de conexión.

Planificación de la gestión del estado

Planning State Management

- **Almacenar datos del lado del cliente:**
  - Cookies.
  - QueryString.
- **Almacenar datos del lado del servidor:**
  - TempData.
  - Application.
  - Session.
  - Bases de datos.

Planificación de la gestión del estado

En el desarrollo de aplicaciones, el estado de la aplicación se refiere a los valores y la información que se mantienen en múltiples operaciones. **HTTP** es fundamentalmente un protocolo sin estado: no tiene mecanismo para retener el estado de la información en varias solicitudes de página. Pero hay muchos escenarios que requieren que **se conserve el estado**, como los siguientes:

- **Preferencias del usuario.**
- **Identidad de usuario.**
- **Carritos de compra.**

Preferencias del usuario

Sitios web que permiten a los usuarios almacenar preferencias. Por ejemplo, el usuario especifica el tamaño preferido para las fotos. Si los valores se pierden entre las solicitudes de página, los usuarios deben aplicar repetidamente la preferencia.

Identidad de usuario

El sitio autentica a usuarios para proporcionar acceso a contenidos exclusivos. Si la identidad del usuario se pierde entre las solicitudes de página, el usuario debe volver a ingresar las credenciales en cada vista.

Carritos de compra

Si el contenido de un carrito de la compra debe mantenerse entre solicitudes de página y asegurar que el cliente puede comprar cualquier cosa desde la aplicación.

En general, se debe **tener cuidado al mantener estado del lado del servidor en grandes cantidades de datos**, porque consume memoria de forma predeterminada. Puedes cambiar el modo de almacenamiento, a fin de evitar problemas de rendimiento.

Almacenamiento de estado del lado del cliente

Cuando se almacena información de estado en el cliente, tener en cuenta que toda la información de estado de ese lado debe enviarse entre el servidor web y el navegador, y este proceso puede ralentizar el tiempo de carga de la página.

Conviene utilizar el almacenamiento de estado del lado del cliente solo para cantidades de datos pequeñas y valores no sensibles.

Cookies

Son pequeños archivos de texto plano que almacenan una pequeña cantidad de valores no sensibles. Los usuarios pueden deshabilitar cookies con fines de privacidad.

QueryString

Es parte de la URL y se usa para comunicar valores de una vista al servidor. Acepta pequeñas cantidades de datos desde una página a otra. No debe usar valores sensibles.

Ejemplo de QueryString:

http://www.contoso.com/default.aspx**?fullname=Emma%20Delip**

QueryString

Almacenamiento de estado del lado del servidor

Se realiza de forma predeterminada en memoria. Las siguientes configuraciones administran el estado en el servidor:

- **TempData**.
- **Application**.
- **Session**.
- **Bases de datos**.

TempData

Se usa en aplicaciones MVC para almacenar valores entre una solicitud y otra.

> Por ejemplo: pasar un mensaje de error a una vista de error.

Application

Se usa para almacenar valores durante la vida útil de la aplicación. Los valores se comparten entre todos los usuarios.

> Por ejemplo: un contador de visitas.

Session

Se usa para almacenar información durante la vida útil de una única sesión de navegador. Los valores almacenados son específicos de la sesión del usuario; no pueden ser accedidos por otros usuarios.

> Por ejemplo: cada usuario configura la apariencia del sitio.

Bases de datos

Se usa una base de datos para almacenar el estado. Este es un buen lugar para almacenar grandes volúmenes de datos que no se pueden colocar en la memoria del servidor o en la computadora del cliente.

> Por ejemplo: almacenar un valor de ID simple en una sesión y usarlo para consultar y actualizar un registro en la base de datos.

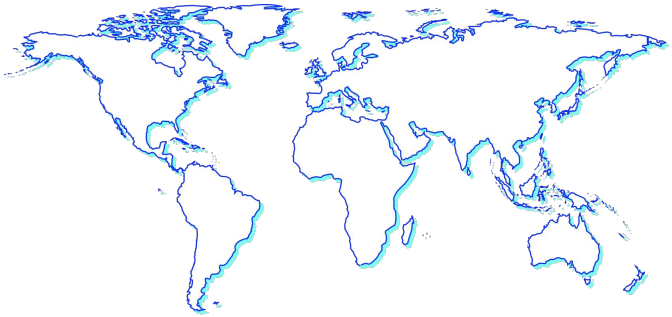
educación IT
CODE YOUR FUTURE

Planificación de la globalización y la localización

Planificación de la globalización y la localización

El proceso que posibilita que **una aplicación web esté disponible en múltiples idiomas** se denomina ***globalización*** o ***internacionalización***. En la planificación de este proceso, se debe contemplar que todos puedan leer el contenido independientemente de la cultura del usuario.

Para presentar contenido personalizado se usan los códigos de idioma reconocidos internacionalmente en los navegadores, se adapta el contenido de la vista al idioma o región del usuario. **Se utilizan vistas independientes para adaptarse a cada código de idioma.**



Planificación de aplicaciones web accesibles

Planificación de aplicaciones web accesibles

Seguir las siguientes pautas, con el objetivo de que el contenido sea accesible para la más amplia gama de usuarios:

- Usar atributos alternativos para contenido visual y auditivo.
- No confiar en el color para resaltar el contenido.
- Separar el contenido de la estructura y el código de presentación:
  - Usar solo tablas para presentar contenido tabular y evitar las tablas anidadas.
  - Usar elementos `<div>` y hojas de estilo posicionales para diseñar elementos en la página.
  - Evitar el uso de imágenes que incluyan texto importante.
  - Ubicar el texto importante en elementos HTML o atributos `alt`.

El Consorcio ***World Wide Web* (*W3C*)** tiene un proyecto llamado ***Web Accessibility Initiative*** (***WAI***) que promueve **contenido web accesible**.

Los usuarios tienen diferentes requisitos según sus habilidades y discapacidades. Por ejemplo:

- Los usuarios con problemas de visión pueden utilizar un navegador estándar, pero tienen la posibilidad de aumentar el tamaño del texto con una lupa.
- Los usuarios invidentes pueden usar un navegador con software de texto a voz o hardware de texto a Braille.
- Los usuarios daltónicos pueden tener dificultades si se usa el color para resaltar el texto.
- Es posible que los usuarios sordos no puedan acceder al contenido de audio.
- Los usuarios con destreza limitada pueden tener dificultades para hacer clic en objetivos pequeños.
- Los usuarios con epilepsia pueden tener convulsiones si se les presenta contenido intermitente.

Puedes encontrar el recurso en este link.

educación IT
CODE YOUR FUTURE

¡Sigamos trabajando!