

Sistemas de base de datos orientada a objetos

UNIDAD TEMÁTICA 2

Nombre	Matricula	Carrera	Porcentaje	Fotografía
Andrea Fernanda Carranza Avalos	1846049	IAS	100%	
Saul Isaías Leija Soriano	1854097	ITS	100%	
Axel Eduardo los Reyes Berrones	1819939	ITS	100%	
Juan Antonio Rodarte Granados	1796397	ITS	100%	
Olympia Briones Peñaloza	2031220	IAS	100%	
Arturo García García	1931808	IAS	100%	
Alfred Valderrabano Pacheco	1805839	IAS	100%	

Índice

Introducción	3
Orientación a objetos	4 – 7
Perspectivas del MOO.....	8 – 10
Base de datos orientada a objetos	11
Relaciones VS Orientada a objetos	12
Ejemplo de base de datos orientada a objetos	13
Ambito	14
Ventajas y Desventajas de las BDOO.....	15
Arquitectura Cliente/Servidor	16 - 19
Arquitectura Multicapas	20 - 22
Cooperación cliente-servidor	23 – 24
Conclusión Grupal	25
Conclusión individual	26 – 27
Referencia	28

Introducción

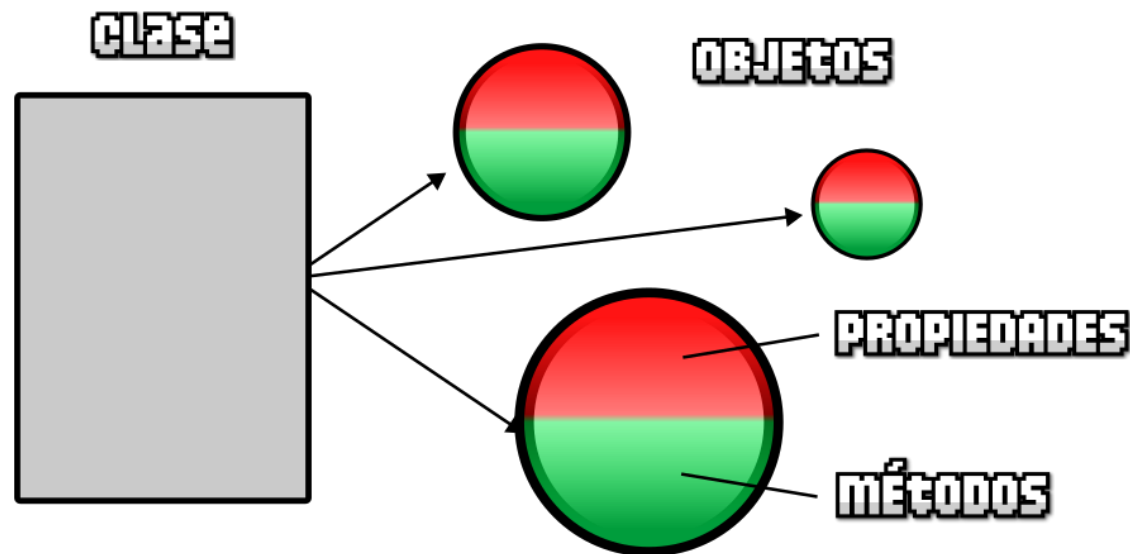
Las bases de datos orientadas a objetos representan una evolución en el almacenamiento y manejo de datos, diseñadas para modelar el mundo real de manera más natural y estructurada. A diferencia de las bases de datos relacionales que utilizan tablas y relaciones, las bases de datos orientadas a objetos utilizan conceptos como clases, objetos y métodos para organizar la información.

Este enfoque permite una representación más fiel de objetos complejos y sus interacciones, facilitando el desarrollo de aplicaciones sofisticadas y eficientes.

Orientación a objetos

La Programación Orientada a Objetos es un paradigma de programación que viene a innovar la forma de obtener resultados.

Está basada en varias técnicas del sexenio: herencia, cohesión, abstracción, polimorfismo, acoplamiento y encapsulamiento.



Conceptos fundamentales



Clase: Una clase es una especie de "plantilla" en la que se definen los atributos y métodos predeterminados de un tipo de objeto



Objeto: Instancia de una clase.



Método: Puede producir un cambio en las propiedades del objeto, o la generación de un "evento".



Evento: Es un suceso en el sistema



Atributo: Características que tiene la clase.



Mensaje: Una comunicación dirigida a un objeto, que le ordena que ejecute uno de sus métodos con ciertos parámetros.

Características

Abstracción: simplifica el sistema mostrando solo los datos esenciales y ocultando detalles complejos para facilitar su uso.

Encapsulamiento: agrupa datos y métodos relacionados, limitando el acceso directo para mejorar la seguridad y la organización.

Polimorfismo: permite a diferentes tipos de datos ser tratados de manera uniforme, facilitando la flexibilidad y reutilización de consultas y operaciones.

Herencia: permite que una tabla herede columnas y restricciones de otra, promoviendo la reutilización y organización de datos.

Modularidad: divide el sistema en módulos independientes que pueden ser desarrollados y mantenidos por separado, mejorando la manejabilidad.

Principio de ocultación: restringe el acceso a ciertos detalles de implementación, mostrando solo lo necesario y mejorando la seguridad.

Recolección de basura: elimina automáticamente los datos no utilizados, optimizando el uso de recursos y evitando sobrecargas.

Modelado orientado a objetos

En el desarrollo de sistemas de software orientado a objetos, el MOO consiste en construir modelos del sistema mediante la identificación y especificación de objetos relacionados que colaboran según los requisitos del sistema.

Perspectivas del MOO

Dimensión estructural de los objetos: Se centra en las propiedades estáticas o pasivas de los sistemas.

Dimensión dinámica del comportamiento: Se centra en las propiedades activas y describe el comportamiento individual y la colaboración entre los objetos.

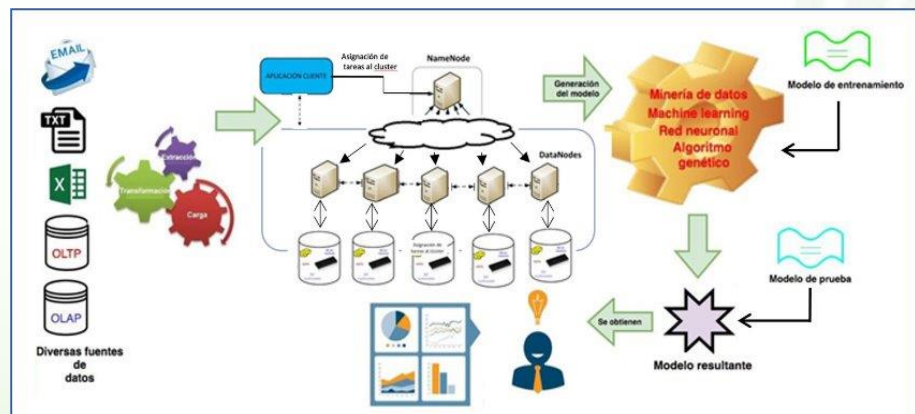
Dimensión funcional de los requerimientos: Son consideradas las propiedades relativas a la función de transformación del sistema de objetos.

PROCESO DEL MOO

Identificar las clases, objetos y atributos:
Determina las clases, objetos y atributos necesarios para el modelo.

Asociar estáticamente los objetos:
Configura una estructura estática que muestre relaciones dependientes del problema.

Describir el comportamiento de los objetos: Especifica el comportamiento de los objetos usando estado, regla de transición, evento y acción.



PROCESO DEL MOO

Definir la colaboración del comportamiento de los objetos: Refleja la interacción entre objetos a través del flujo de eventos o mensajes.

Organizar las clases en jerarquías de herencia: Organiza las clases para maximizar la compartición de propiedades comunes.

Agregar y/o particionar las clases por niveles de abstracción: Jerarquiza el modelo por niveles de complejidad usando particionamiento o agregación.

BASE DE DATOS ORIENTADA A OBJETOS

Un Sistema de Gestión de Bases de Datos Orientado a Objetos (ODBMS) combina las características de una base de datos con las de un lenguaje de programación orientado a objetos. Esto permite manejar datos persistentes, control de concurrencia, recuperación de datos y consultas asociativas de manera transparente.

Relaciones VS Orientada a objetos

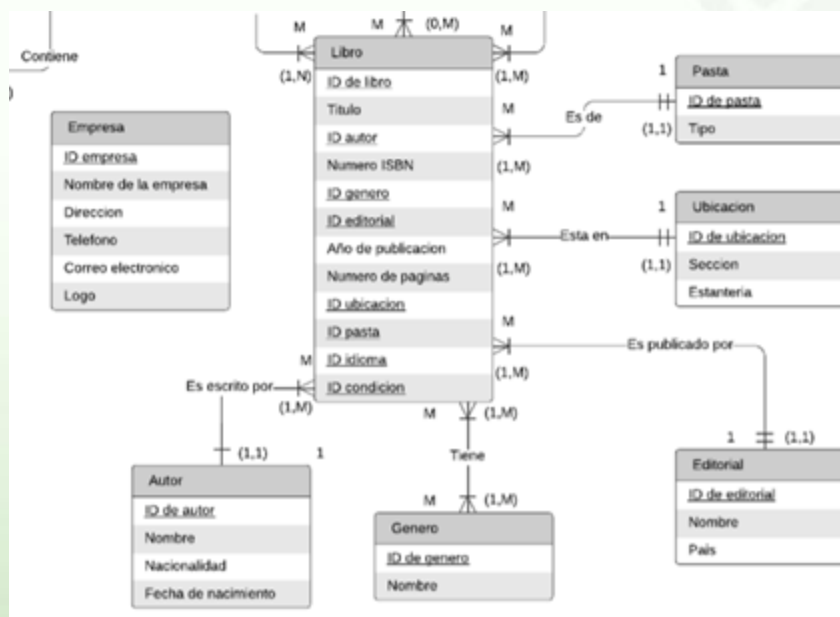
Las base de datos relacionales son el estándar en programación y desarrollo web desde hace mucho tiempo. En este modelo, **la información se almacena en tablas relacionadas entre sí**. Las relaciones también permiten almacenar y consultar información compleja compuesta por varios elementos.

Todos los atributos de cada objeto están disponibles de inmediato y, además, los registros pueden ser mucho más complejos

Ejemplo de base de datos orientada a objetos

Libreria

Entidades principales:



Estos objetos pueden estar interrelacionados de manera que un libro puede estar asociado con múltiples préstamos y cada préstamo está vinculado a una persona específica.

Ámbito

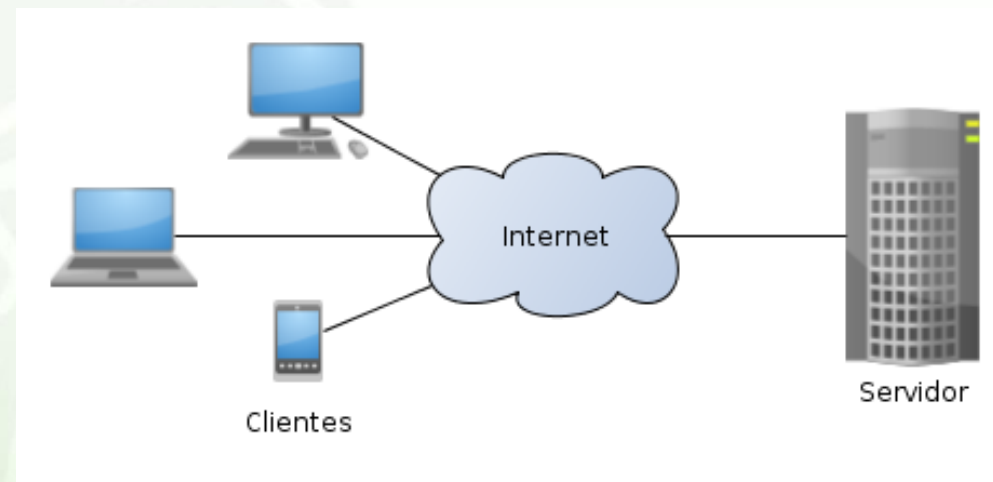
Las bases de datos orientadas a objetos (ODBMS) están diseñadas para trabajar con lenguajes de programación orientados a objetos como Java, C#, Visual Basic.NET y C++. Son ideales para sistemas que necesitan buen rendimiento en la manipulación de datos complejos, ya que almacenan objetos en disco y se integran de manera transparente con el programa, reduciendo los costos de desarrollo y mantenimiento.

Ventajas y Desventajas de las BDOO

Ventajas	Desventajas
Aseguran un alto nivel de confiabilidad mediante la ejecución de transacciones ACID, garantizando que las transacciones solo se completen sin conflictos.	Aún no han alcanzado una madurez plena en su desarrollo, lo que aumenta el riesgo de sostenibilidad a largo plazo.
Cuentan con un mecanismo de almacenamiento en caché que crea réplicas parciales de la base de datos, minimizando el acceso al disco y el tráfico de red.	Poca experiencia y documentación disponible.

Arquitectura cliente-servidor

La arquitectura cliente-servidor distribuye tareas entre servidores (proveedores de recursos) y clientes (demandantes de servicios). Los clientes realizan peticiones y los servidores responden. Es ventajosa en sistemas operativos multiusuario en red, pero también aplicable a programas en una sola computadora.



Características cliente

Es quien inicia solicitudes o peticiones, tienen por tanto un papel activo en la comunicación (dispositivo maestro o amo).

Espera y recibe las respuestas del servidor.

Por lo general, puede conectarse a varios servidores a la vez.

Normalmente interactúa directamente con los usuarios finales mediante una interfaz gráfica de usuario.

Características servidor

Al iniciarse esperan a que lleguen las solicitudes de los clientes, desempeñan entonces un papel pasivo en la comunicación (dispositivo esclavo).

Tras la recepción de una solicitud, la procesan y luego envían la respuesta al cliente.

Por lo general, acepta las conexiones de un gran número de clientes (en ciertos casos el número máximo de peticiones puede estar limitado).

Características generales de su arquitectura

Cliente y servidor pueden ser entidades separadas o una sola.

Funciones de cliente y servidor pueden estar en plataformas separadas o en la misma y escalarse independientemente.

Los cambios en las plataformas no afectan la interrelación entre hardware y software.

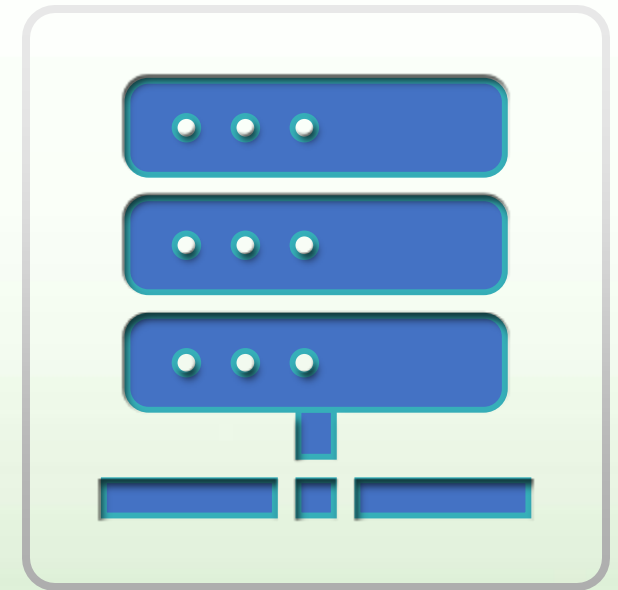
Infraestructura oculta la complejidad de datos y protocolos, facilitando el acceso a recursos de la red.

El usuario suele tener aplicaciones y bases de datos en su PC, sin depender del sistema central de la organización.

Arquitecturas multicapas

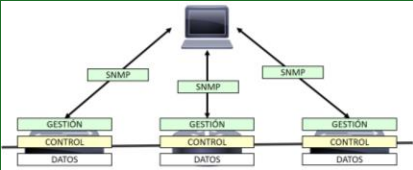
La arquitectura cliente/servidor genérico tiene dos tipos de nodos: clientes y servidores.

- Clientes que interactúan con usuarios finales.
- Servidores de aplicación que procesan datos para los clientes.
- Servidores de base de datos que almacenan datos para los servidores de aplicación.



Ventajas de las BDOO

Centralización de control



Escalabilidad



Fácil mantenimiento



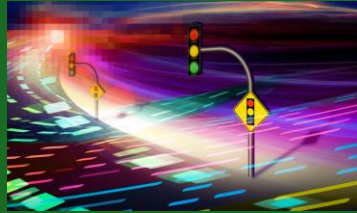
Seguridad y facilidad



Privacidad



Congestión del tráfico



Dependencia del
servidor



Desventajas de las BDOO

Limitación de recursos




Restricción de
información




Cooperación cliente-servidor

Múltiple Server: Utilizar múltiples terminales para realizar la misma tarea, mejorando la eficiencia mediante la centralización de recursos y mayor hardware/software.



Cooperación de procesos paralelos: Ejecutar el mismo proceso simultáneamente en sistemas redundantes.



Cooperación de base de datos: Aprovechar la información ya existente en lugar de crearla de nuevo, interactuando con los datos disponibles.

Presentación/Captación
de la información.

Procesos.

Almacenamiento de la
información.

Puestos de trabajo

Comunicaciones.

Conclusión General

Las ODBMS representan una alternativa valiosa a las bases de datos relacionales tradicionales, ofreciendo ventajas significativas en términos de integración con lenguajes de programación orientados a objetos y flexibilidad en la representación de datos complejos.

Por otro lado, la arquitectura cliente-servidor sigue siendo una estructura fundamental en el diseño de sistemas distribuidos, especialmente en el contexto de aplicaciones web y servicios en línea. Esta arquitectura facilita la distribución de tareas entre múltiples servidores y clientes, optimizando el uso de recursos y mejorando la escalabilidad y la eficiencia de las aplicaciones.

Conclusiones Individuales

Saul Isaías Leija Soriano 1854097

Gracias a este capítulo se logra entender cuales son las diferentes tipos de base de datos, abarca mucho de las orientada a objetos y esto nos ayuda a tener una perspectiva más grande de cómo escoger una base de dato dependiendo que quiera el cliente

Axel Eduardo de los Reyes Berrones 1819939

La idea de una ODBMS fue un algo nuevo que aprendí en lo que llevo de mi carrera, por lo general estamos demasiado acostumbrados a las bases de datos relacionales que no logras ver todas las ventajas que te puede ofrecer una diferente y verlo como un tema fue de mucho beneficio. Otro de los temas que abordamos fue el de cliente servidor, el cual es una arquitectura sumamente importante ya que es una de las más comunes al usar el internet.

Olympia Briones Peñaloza 2031220

Los sistemas de bases de datos orientadas a objetos están hechos para almacenar datos de manera que se parezcan a cosas reales como objetos. Esto hace más fácil trabajar con información complicada y estructurada. Sin embargo, usarlos bien requiere entender cómo organizar los datos para que la base de datos funcione bien y la información esté siempre correcta y disponible cuando se necesita.

Conclusiones individuales

Andrea Fernanda Carranza Avalos 1846049

Como conclusión creo que los sistemas de base de datos orientados a objetos ofrecen ventajas significativas, aunque han enfrentado desafíos en términos de rendimiento y escalabilidad en comparación con los sistemas racionales, continúan siendo una buena opción para aplicaciones específicas que requieren flexibilidad y eficiencia en el manejo de datos complejos y estructuras de objetos.

Arturo García García 1931808

En este capítulo, se abordaron dos modelos más de administrar bases de datos. Por un lado tenemos el modelo de bases de datos orientado a objetos, este modelo trabaja con los mismos conceptos de la programación orientado a objetos, como lo son clase, atributo, objeto, etc. Con el cual se trabajan con los lenguajes orientados a objetos como el php, java, y python. En el modelo cliente/servidor, podemos concluir que es el modelo con el más se le compara al de bases distribuidas, siendo este centralizado a un solo nodo a diferencia del distribuido que eran en varios nodos, viendo como existe la colectividad entre el cliente y servidor a través del middleware.

Juan Antonio Rodarte Granados 1796397

En este capítulo, hemos aprendido cómo el control de concurrencia es vital para el manejo eficiente de bases de datos. Este concepto nos ayuda a asegurar que múltiples transacciones se ejecuten sin interferencias, manteniendo la integridad de los datos. También exploramos las bases de datos distribuidas, que son esenciales hoy en día para manejar grandes cantidades de usuarios simultáneamente en proyectos de software.

Conclusiones individuales

Alfred Valderrabano Pacheco 1805839

Llegando al final de ésta actividad, podemos concluir que podemos tener mas en claro los tipos de bases de datos, como trabajan uno contra el otro, ventajas, desventajas y saber cuando deben ser empleadas, las bases de datos orientadas a objetos son un herramienta muy poderosa para trabajar con información mas compleja y difícil, por ultimo se tocó el modelo cliente/servidor, que creo yo no tuvimos problema con ese tema debido a que es un modelo con el que nos topamos en la vida cotidiana y sabemos identificarlo.

Referencias

- Coad, P; Yourdon, E. (1991). Prentice-Hall International editions, ed. Object-oriented Design. ISBN 9780136300700.
- Kim, Won. Introduction to Object-Oriented Databases. The MIT Press, 1990. ISBN 0-262-11124-1
- Tema 2. Bases de datos orientadas a objetos Diseño de Sistemas de Bases de Datos. Merche Marques