

Bases de datos orientadas a objetos

Ricardo Pérez López

IES Doñana, curso 2025/2026

Generado el 2026/01/01 a las 22:40:00

Índice

1. Introducción a las bases de datos orientadas a objetos (OODB)	2
1.1. Conceptos clave	2
1.1.1. Objetos, clases y herencia	2
1.1.2. Persistencia de objetos	2
1.1.3. Integración de datos y comportamiento	2
1.2. Comparación con RDBMS	2
1.2.1. Modelos de datos relacionales vs. orientados a objetos	2
1.2.2. Ventajas y desventajas de OODB	2
1.2.3. Escenarios donde es preferible usar una OODB	2
1.3. Casos de uso comunes	2
1.3.1. Aplicaciones de simulación	2
1.3.2. Modelos de datos complejos	2
1.3.3. Aplicaciones con estructuras de datos jerárquicas	2
2. Persistencia de objetos en Python	2
2.1. Serialización y deserialización de objetos (módulo <code>pickle</code>)	2
3. Conceptos de bases de datos orientadas a objetos	2
3.1. Modelado de una OODB	2
3.1.1. Clases como tablas	2
3.1.2. Relaciones entre objetos (1 a 1, 1 a muchos, muchos a muchos)	2
3.1.3. Polimorfismo en la base de datos	2
3.2. Implementación básica en Python	2
3.2.1. Guardar objetos en una estructura persistente	3
3.2.2. Simular una base de datos orientada a objetos sencilla	3
3.3. Limitaciones de una implementación casera	3
3.3.1. Consistencia de datos	3
3.3.2. Concurrencia	3
3.3.3. Integridad referencial	3
4. Introducción a Zope Object Database (ZODB)	3
4.1. ¿Qué es ZODB?	3

4.1.1. Ventajas de usar ZODB	3
4.1.2. Comparación con otras soluciones de bases de datos	3
4.2. Instalación y configuración	3
4.2.1. Estructura básica de un proyecto con ZODB	3
4.3. Persistencia de objetos en ZODB	3
4.3.1. Crear una base de datos en ZODB	3
4.3.2. Guardar, actualizar y eliminar objetos	3
4.3.3. Administración de la base de datos y control de versiones	3

1. Introducción a las bases de datos orientadas a objetos (OODB)

1.1. Conceptos clave

- 1.1.1. Objetos, clases y herencia
- 1.1.2. Persistencia de objetos
- 1.1.3. Integración de datos y comportamiento

1.2. Comparación con RDBMS

- 1.2.1. Modelos de datos relacionales vs. orientados a objetos
- 1.2.2. Ventajas y desventajas de OODB
- 1.2.3. Escenarios donde es preferible usar una OODB

1.3. Casos de uso comunes

- 1.3.1. Aplicaciones de simulación
- 1.3.2. Modelos de datos complejos
- 1.3.3. Aplicaciones con estructuras de datos jerárquicas

2. Persistencia de objetos en Python

2.1. Serialización y deserialización de objetos (módulo `pickle`)

3. Conceptos de bases de datos orientadas a objetos

3.1. Modelado de una OODB

- 3.1.1. Clases como tablas
- 3.1.2. Relaciones entre objetos (1 a 1, 1 a muchos, muchos a muchos)
- 3.1.3. Polimorfismo en la base de datos

3.2. Implementación básica en Python

- 3.2.1. Guardar objetos en una estructura persistente
- 3.2.2. Simular una base de datos orientada a objetos sencilla
- 3.3. Limitaciones de una implementación casera
 - 3.3.1. Consistencia de datos
 - 3.3.2. Concurrencia
 - 3.3.3. Integridad referencial
- 4. Introducción a Zope Object Database (ZODB)
 - 4.1. ¿Qué es ZODB?
 - 4.1.1. Ventajas de usar ZODB
 - 4.1.2. Comparación con otras soluciones de bases de datos
 - 4.2. Instalación y configuración
 - 4.2.1. Estructura básica de un proyecto con ZODB
 - 4.3. Persistencia de objetos en ZODB
 - 4.3.1. Crear una base de datos en ZODB
 - 4.3.2. Guardar, actualizar y eliminar objetos
 - 4.3.3. Administración de la base de datos y control de versiones