

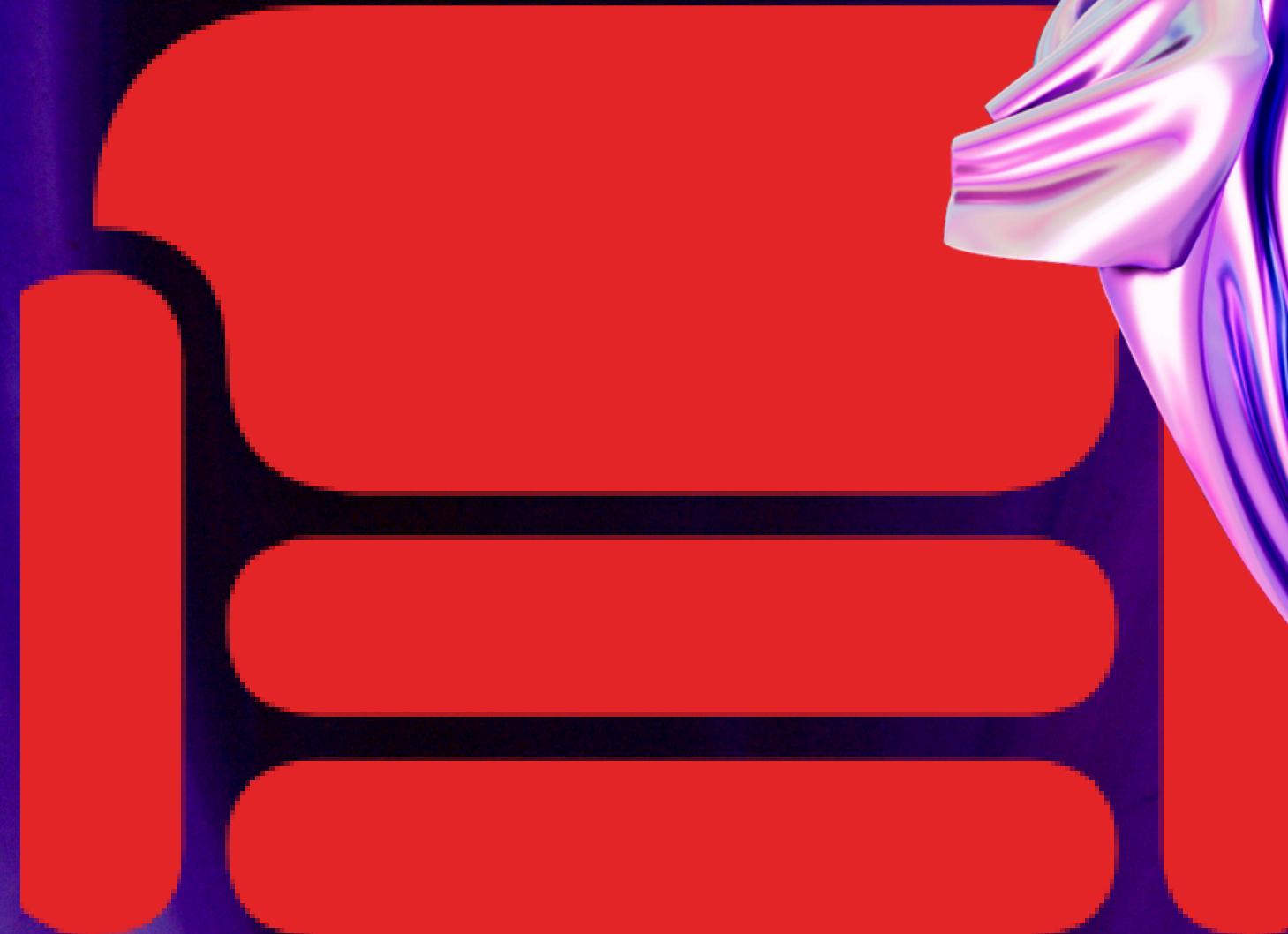
COUCHDB

Elaborado por:
Gonzalo Santiago Garcia

Indice

- **¿Qué es Couchdb?**
- **Historia**
- **Características**
- **Beneficios Principales**
- **Arquitectura de CouchDB**
- **Ventajas**
- **Desventajas**
- **Tabla comparativa**
- **Conclusión**
- **Referencias**

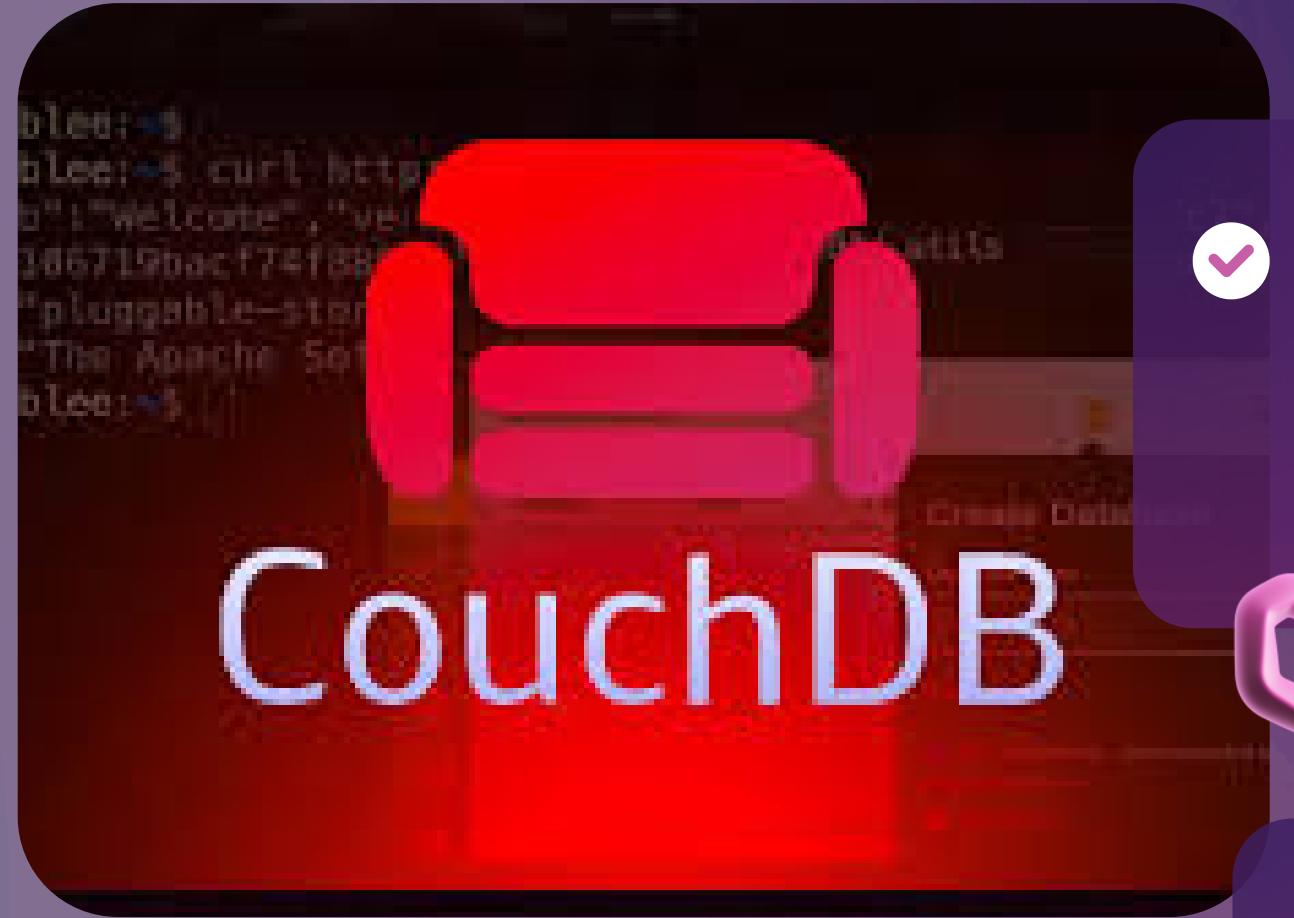
¿Qué es Couchdb?



CouchDB rela'

Es una base de datos NoSQL que almacena información en documentos JSON, lo que permite manejar datos flexibles sin necesidad de un esquema fijo.





Historia

- ✓ Fue creada en 2005 y se integró al ecosistema de Apache en 2008, lo que garantiza un desarrollo continuo y una comunidad activa.
- ✓ Está diseñada para ser fácil de usar, resistente a fallos y accesible desde prácticamente cualquier entorno.
- ✓ Su enfoque está orientado a la web moderna, dispositivos móviles y aplicaciones distribuidas.



Características



Permite sincronizar bases de datos entre múltiples servidores, dispositivos y ubicaciones.

La replicación funciona automáticamente detectando cambios en cualquier copia y actualizándolos en las demás.

Esto permite:

- Funcionamiento en sistemas distribuidos
- Alta disponibilidad
- Menos riesgo ante fallas del servidor
- Copias independientes que siguen sincronizadas
- Ideal para apps que deben funcionar tanto en la nube como localmente.



Vistas y MapReduce

- CouchDB usa **vistas** para consultar datos de manera flexible. Son funciones que transforman documentos en resultados filtrados o estructurados.
- No alteran los documentos originales, lo que permite tener infinitas vistas según la necesidad del proyecto.
- Usa MapReduce, lo que permite:
- Crear resúmenes rápidos de grandes cantidades de datos
- Obtener resultados instantáneos incluso con millones de documentos
- Esto hace que CouchDB sea muy eficiente en análisis de información distribuida.

API HTTP (REST)



Q1:

- Toda la base de datos se maneja mediante solicitudes HTTP, como si fuera una página web.



Q2:

- Esto permite que cualquier lenguaje o plataforma pueda conectarse sin configuraciones complicadas.



Q3:

Soporta fácilmente operaciones CRUD:

- Crear
- Leer
- Actualizar
- Eliminar



Q4:

- Es extremadamente accesible para desarrolladores modernos



Beneficios Principales



1.- Excelente escalabilidad

- Puede distribuir datos entre múltiples nodos mediante:
 - Partición horizontal
 - Replicación simultánea
- Su arquitectura permite manejar tanto pequeñas apps como sistemas empresariales enormes.

2.- Alto rendimiento sin bloqueos

- Utiliza MVCC (Multi-Version Concurrency Control), lo que evita bloquear datos durante lecturas y escrituras.
- Esto permite:
 - Lecturas más rápidas
 - Menos conflictos
 - Mejor rendimiento bajo alta carga
- Solo se rechazan actualizaciones si hay un conflicto real simultáneo.

3. Proyecto de código abierto

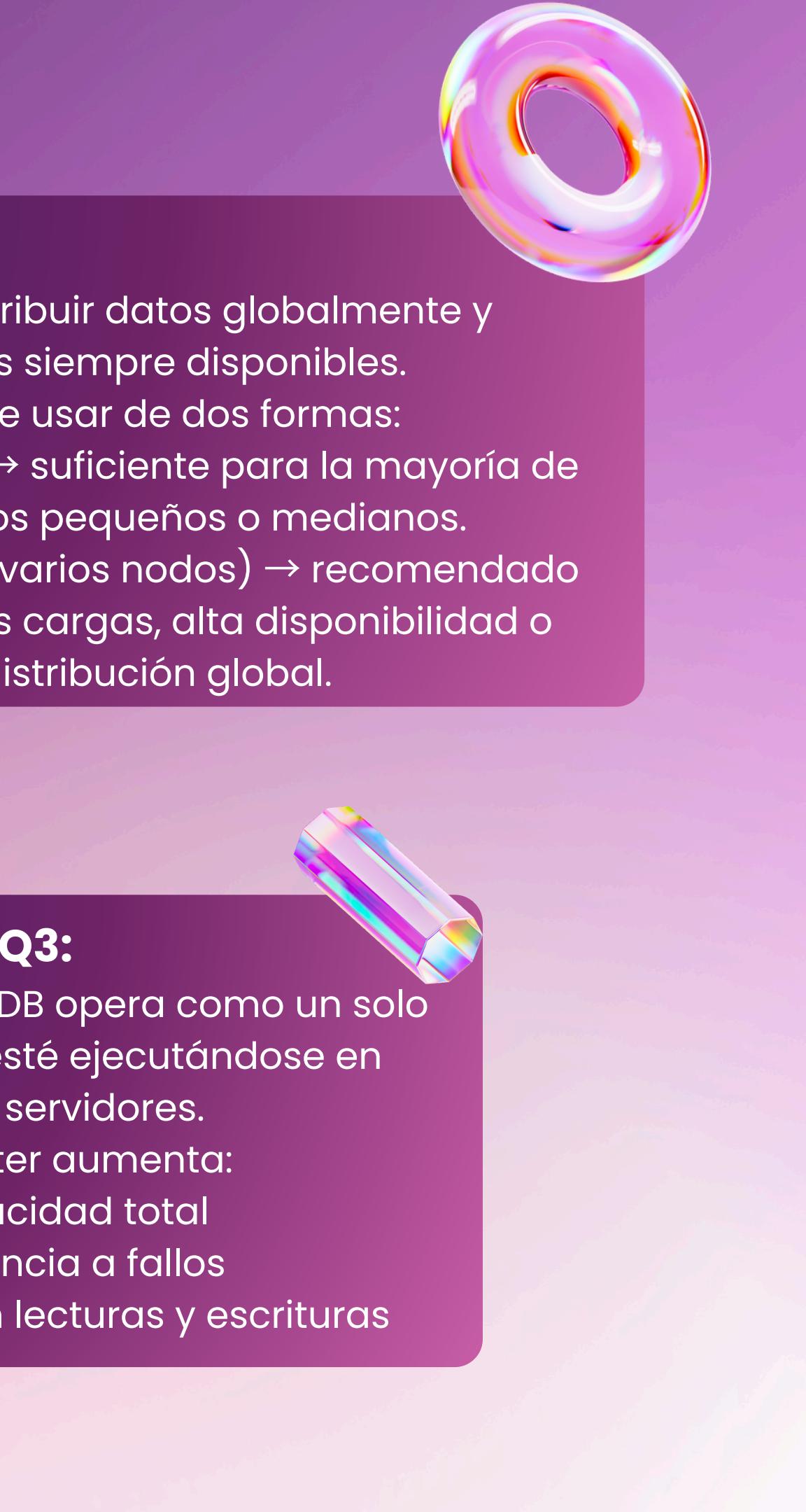


- Tiene una comunidad global que lo mejora constantemente.
- Ofrece una estabilidad confiable similar a productos comerciales, pero sin costos ni restricciones.
- Ideal para empresas que buscan longevidad en la tecnología.

Arquitectura de CouchDB

Q1:

- CouchDB está diseñado para ser escalable gracias a su arquitectura distribuida.
- Utiliza replicación multimaster, donde varios nodos pueden recibir escrituras y sincronizarse entre sí.



Q2:

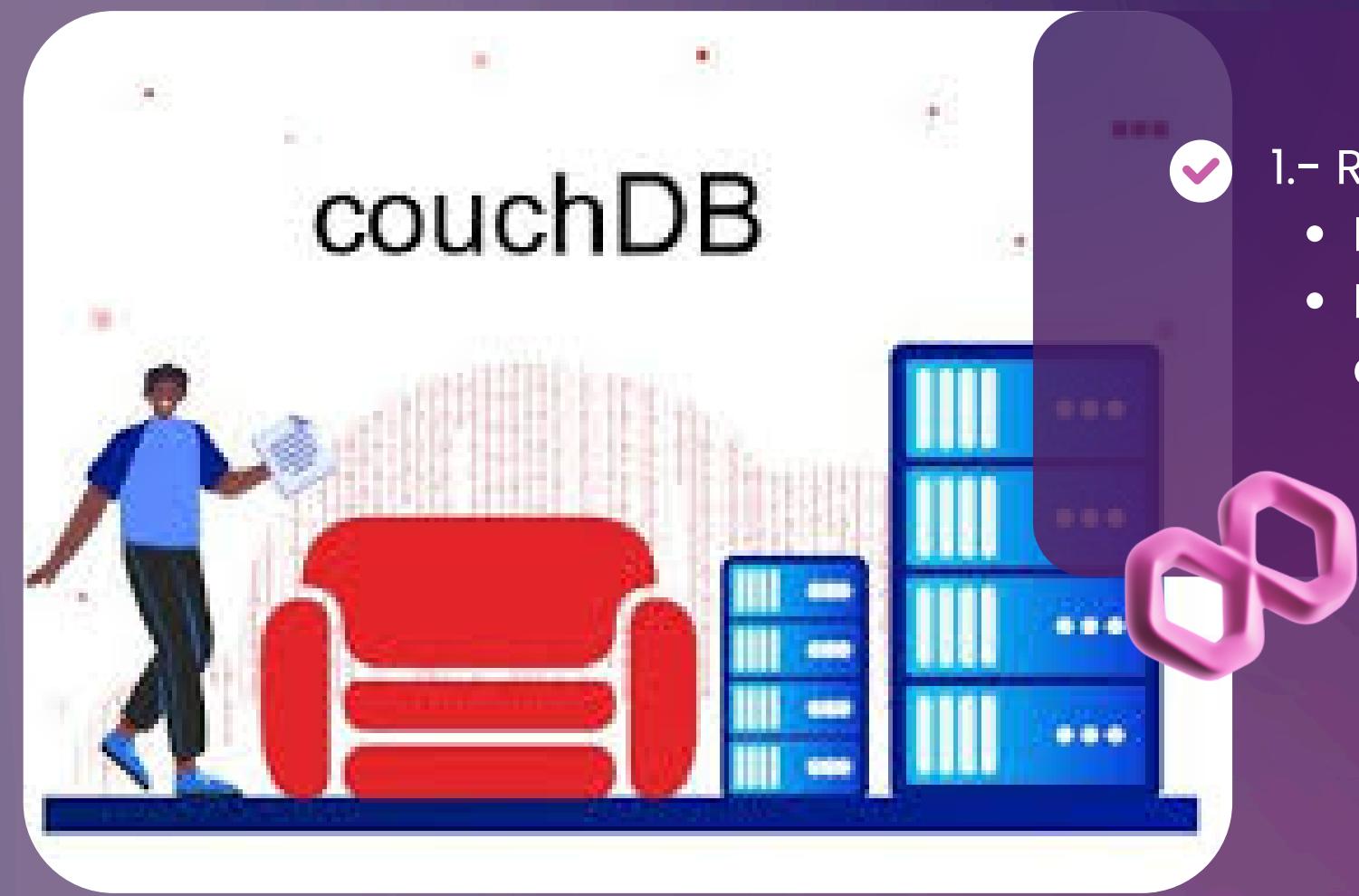
Esto permite distribuir datos globalmente y mantenerlos siempre disponibles.

- Se puede usar de dos formas:
 - Un solo nodo → suficiente para la mayoría de proyectos pequeños o medianos.
 - Modo clúster (varios nodos) → recomendado para grandes cargas, alta disponibilidad o distribución global.

Q3:

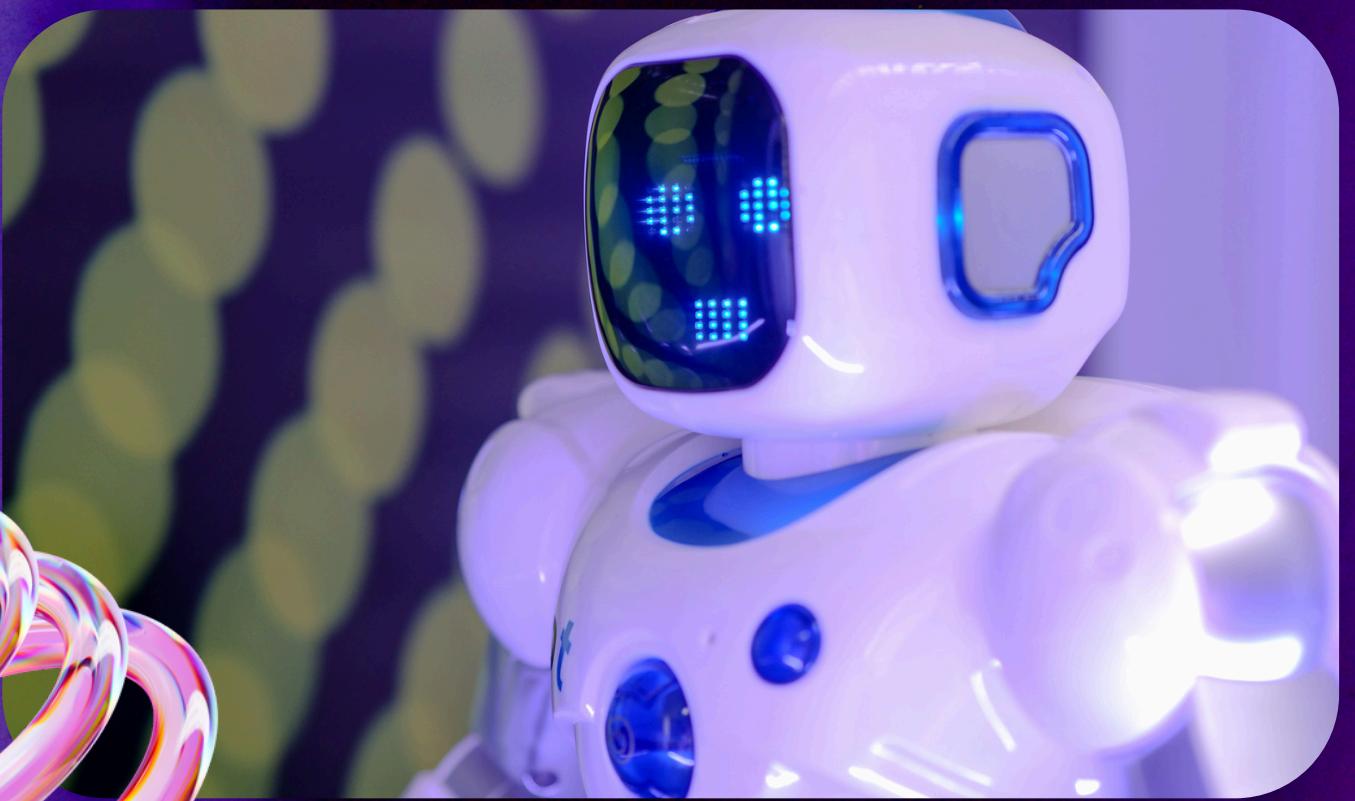
En un clúster, CouchDB opera como un solo servicio, aunque esté ejecutándose en muchos servidores.

- El clúster aumenta:
 - Capacidad total
 - Tolerancia a fallos
 - Desempeño en lecturas y escrituras



Ventajas

- 1.- Replicación offline/online muy robusta
 - Perfecto para apps móviles o web offline-first.
 - Los dispositivos pueden trabajar sin conexión y sincronizar datos automáticamente cuando vuelven a conectarse.
- 2.- Escalabilidad distribuida
 - Permite tener nodos en diferentes regiones del mundo.
 - Cada nodo puede funcionar de forma independiente.
 - La sincronización automática mantiene todos los datos coherentes.
- 3.- Modelo de datos flexible
 - Usa documentos JSON sin esquema obligatorio.
 - Los campos pueden cambiar, crecer o adaptarse sin modificar estructuras existentes.



4. Alta concurrencia sin bloqueos

- Gracias a MVCC, múltiples usuarios pueden leer y escribir al mismo tiempo.
- No bloquea documentos ni tablas, lo que aumenta el rendimiento en cargas altas.

5. Basado en estándares web

- Su API REST/HTTP facilita integraciones con JavaScript, aplicaciones web, microservicios, etc.



Desventajas

1. Consistencia eventual

- Los datos pueden tardar en reflejarse en todos los nodos.
- No es ideal para sistemas que requieren transacciones críticas e instantáneas (bancos, pagos, etc.).

2. Necesita aprender MapReduce

- Las consultas avanzadas se hacen con MapReduce en JavaScript.
- Requiere curva de aprendizaje para crear vistas e índices eficientes.

3. Consultas complejas menos eficientes

- No está optimizado para joins o consultas relacionales.
- Puede ser más lento que SQL o bases NoSQL con búsqueda avanzada.

4. Ecosistema más pequeño

- Menor adopción que MongoDB o Couchbase.
- Menos herramientas, plugins o soporte empresarial.

Tabla comparativa

Característica	CouchDB	Couchbase
Tipo de base de datos	NoSQL orientada a documentos	NoSQL multimodelo (documentos, clave-valor)
Formato de datos	JSON	JSON
Lenguaje de consulta	MapReduce y Mango (similar a JSON Query)	N1QL (similar a SQL)
Arquitectura	Replicación multi-master	Arquitectura distribuida con clústeres
Escalabilidad	Escala horizontal mediante replicación	Escalabilidad horizontal automática
Replicación	Replicación bidireccional y offline-first	Replicación entre nodos y centros de datos
Manejo de conflictos	Resolución automática o manual	Manejo interno con alta disponibilidad
Consistencia	Eventual	Eventual con opciones de mayor consistencia
Rendimiento	Bueno para cargas moderadas	Alto rendimiento y baja latencia
Uso offline	Muy fuerte (sincronización local)	Limitado comparado con CouchDB
Casos de uso comunes	Aplicaciones móviles, sincronización de datos, sistemas distribuidos	Aplicaciones empresariales, big data, tiempo real
Facilidad de uso	Simple y fácil de configurar	Más complejo, pero más potente
Licencia	Open Source (Apache)	Community Edition (gratis) y Enterprise (pago)
Lenguaje principal	Erlang	C++

Conclusión

CouchDB ofrece una arquitectura distribuida ideal para aplicaciones offline-first, con escalabilidad y replicación avanzada, aunque sacrifica algo de rendimiento en consultas complejas y consistencia inmediata.





Referencias

<https://www.ibm.com/mx-es/think/topics/couchdb>

<https://aprenderbigdata.com/couchdb/> [https://inchoo-](https://inchoo-net.translate.goog/dev-talk/couchdb-for-php-developers-crud/?_x_tr_sl=en&_x_tr_tl=es&_x_tr_hl=es&_x_tr_pto=tc)
[net.translate.goog/dev-talk/couchdb-for-php-developers-](https://inchoo-net.translate.goog/dev-talk/couchdb-for-php-developers-crud/?_x_tr_sl=en&_x_tr_tl=es&_x_tr_hl=es&_x_tr_pto=tc)
[crud/?_x_tr_sl=en&_x_tr_tl=es&_x_tr_hl=es&_x_tr_pto=tc](https://inchoo-net.translate.goog/dev-talk/couchdb-for-php-developers-crud/?_x_tr_sl=en&_x_tr_tl=es&_x_tr_hl=es&_x_tr_pto=tc)