# Bases de Datos en Python

Python es un lenguaje muy flexible que se adapta fácilmente a distintos entornos de desarrollo. Gracias a su gran comunidad y a la variedad de librerías disponibles, es posible trabajar con bases de datos relacionales, no relacionales, locales y en la nube.

Bases de datos relacionales:

* Organizan los datos en tablas (filas y columnas).
* Cada tabla representa una entidad (por ejemplo, usuarios, productos).
* Las relaciones entre datos se hacen mediante claves primarias y foráneas.
* Usan SQL (Structured Query Language) para gestionar datos.
* Ejemplos: MySQL, PostgreSQL, SQLite.

Bases de datos no relacionales (NoSQL):

* No usan tablas tradicionales.
* Pueden almacenar datos como: Documentos (JSON) → MongoDB Clave-valor → Redis Columnas → Cassandra Grafos → Neo4j
* Son más flexibles y se adaptan mejor a datos no estructurados o en constante cambio.
* Ideales para: apps en tiempo real, big data, redes sociales, chats.

Principio del formulario

Final del formulario

Veamos una comparación de todas las bases de datos más utilizadas:

* SQLite: es una base de datos relacional, liviana, que no necesita un servidor. Se guarda en un archivo local, por lo que es ideal para proyectos pequeños, prototipos o pruebas locales. En Python se puede usar fácilmente con la librería sqlite3, que ya viene incluida.
* PostgreSQL: es una base de datos relacional de código abierto, muy robusta y escalable. Es perfecta para aplicaciones complejas, como sistemas empresariales o análisis de datos. Python se conecta a ella con librerías como psycopg2 o SQLAlchemy.
* MySQL y MariaDB: también son relacionales y muy comunes en páginas web y sistemas e-commerce. Son conocidas por su eficiencia y facilidad de uso. Podemos trabajar con ellas desde Python usando mysql-connector-python o PyMySQL.
* MongoDB: es una base de datos NoSQL que guarda los datos como documentos en formato JSON. Es útil para aplicaciones que manejan datos no estructurados, como redes sociales o apps en crecimiento. En Python se usa con la librería PyMongo.
* Firebase (Firestore): es una base de datos NoSQL en la nube. Funciona muy bien con aplicaciones móviles o web en tiempo real, como chats o juegos. Se integra fácilmente con Python a través de firebase-admin.
* Redis: es una base de datos en memoria tipo clave-valor. Es extremadamente rápida y se usa mucho para caché, sesiones de usuario o colas de tareas. En Python se accede mediante redis-py.

Como podemos ver, cada tipo de base de datos tiene su propósito. Python no solo las soporta, sino que ofrece herramientas eficientes para interactuar con ellas según las necesidades del proyecto.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Base de Datos | Tipo | Uso Principal | Ventajas | Librería en Python |
| SQLite | Relacional, embebida | Aplicaciones pequeñas, prototipos, pruebas locales | No requiere servidor, se guarda en un archivo .db | sqlite3 |
| PostgreSQL | Relacional, código abierto | Apps web complejas, análisis de datos | Robusta, escalable, soporte para JSON | psycopg2, SQLAlchemy |
| MySQL / MariaDB | Relacional | Sitios web, e-commerce | Popular, compatible con múltiples lenguajes | mysql-connector-python, PyMySQL |
| MongoDB | NoSQL, documentos (JSON) | Apps con datos no estructurados o en crecimiento | Flexible, no necesita esquemas fijos | PyMongo |
| Firebase (Firestore) | NoSQL en la nube | Apps móviles, juegos, chats, tiempo real | Base de datos + hosting + auth en tiempo real | firebase-admin |
| Redis | Clave-valor en memoria | Caché, sesiones, colas, apps de alto rendimiento | Extremadamente rápida, ideal para datos temporales | redis-py |