# ACTIVIDAD GUIADA: DISEÑO DE BASE DE DATOS NOSQL EN MONGODB


- David Aparicio
- David Camelo

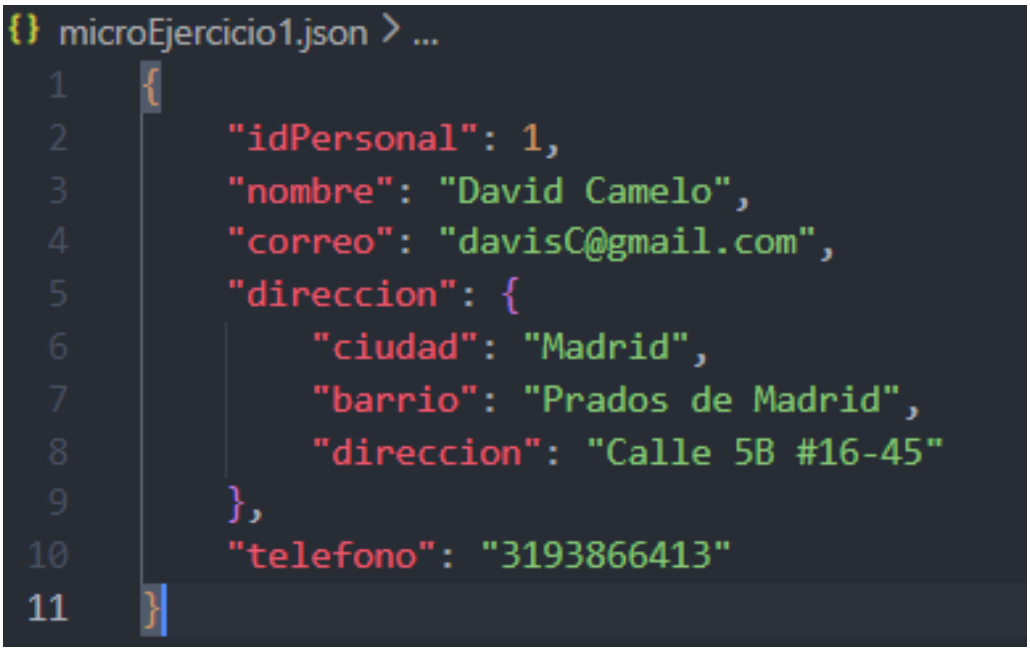
---

**Entrega 1: Diferencias y JSON personal**

Diferencia entre tabla (SQL) y colección (NoSQL / MongoDB)

- **Tabla (SQL):** tienen una estructura rígida con filas y columnas, su esquema fijo y se relacionan las entidades mediante llaves primarias y foráneas, y también debe realizarse normalización.
- **Colección (MongoDB):** conjunto de documentos JSON flexibles donde cada documento puede tener distinto esquema y es ideal para consultas donde se necesita leer documentos completos, también permite embedding para lecturas rápidas y referencing para datos compartidos (similar a las relaciones en las tablas SQL).

**JSON personal sencillo:**




---

**Entrega 2: Colecciones del caso y campos**

**Colección: Clientes**

- _id
- nombre
- correo
- telefono

- direccion (subdocumento: ciudad, barrio, direccion)
- fecha_registro

**Colección: Productos**

- _id
- nombre
- descripcion
- precio
- categoria
- stock
- imagenes (array)

**Colección: Pedidos**

- _id
- cliente_id (referencia)
- fecha
- items (array de subdocumentos: producto_id, cantidad, precio_unitario)
- total
- estado (ej. "pendiente", "enviado", "entregado")

---

**Entrega 3: Embedding vs Referencing (decisiones y justificación)**

**Decisiones:**

- **Clientes**: embebido -> direcciones dentro del documento direccion.
  **Justificación**: la dirección se consulta casi siempre junto con el cliente la lectura única favorece embedding.
- **Productos**: datos principales embebidos y no referenciar stock/imagenes.
  **Justificación**: ficha de producto debe ser autocontenida para mostrar rápidamente.
- **Pedidos**: referencing -> cliente_id y items.producto_id como referencias.
- **Justificación**: pedidos consultan información del cliente y del producto por separado y puede haber muchos pedidos que referencian el mismo cliente y producto; referencing evita duplicar datos y facilita actualizaciones de producto.
- **Items dentro de Pedidos:** embebidos como subdocumentos (cantidad y precio_unitario).
- **Justificación**: los items son parte integral del pedido y se consultan con él.

**RT**: Embedding cuando los datos se leen juntos y la redundancia es aceptable; Referencing cuando los datos se comparten entre muchos documentos o cambian frecuentemente pero estas referenciadas.

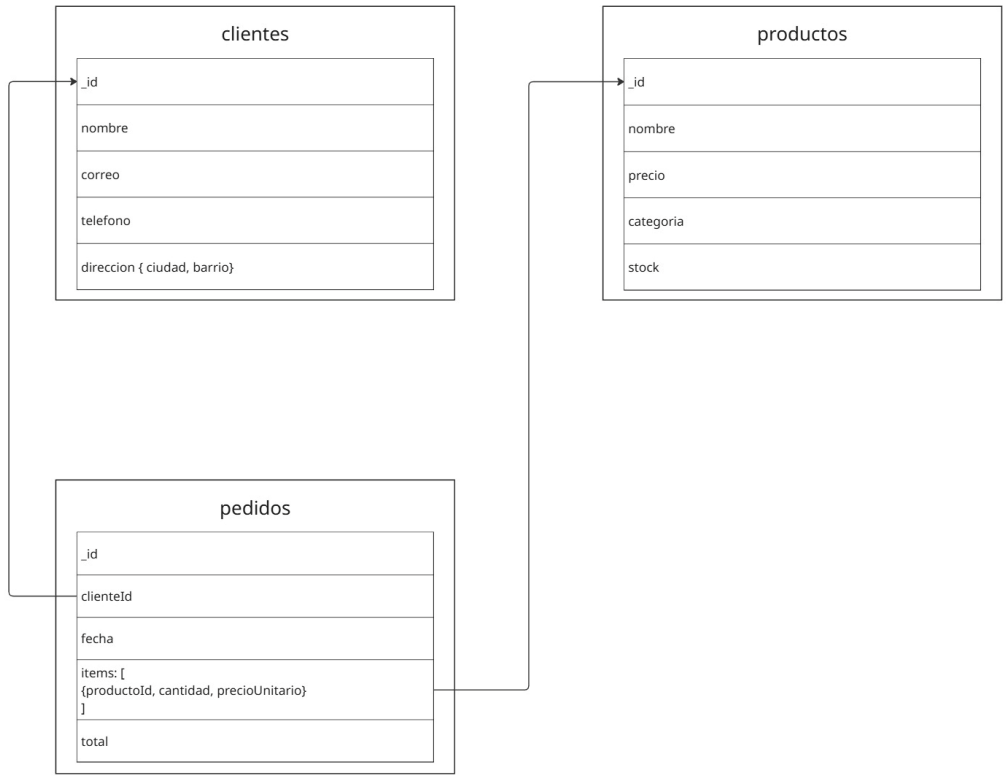
---

**Entrega 4: JSON finales de las colecciones (modelo completo)**

Los archivos JSON finales con las colecciones están en la carpeta subida como “Entrega 4”.

---

**Entrega 5 — Diagrama NoSQL**




---

**Trabajo Individual del Aprendiz – Desarrollo Autónomo**

**Crear tus propias colecciones:**

**Dominio**: Tienda de Mascotas

**Nuevas colecciones:** Mascotas, Servicios, Citas

**Definir documentos JSON reales:** Cada documento incluye: _id, atributos clave, subdocumento, arreglo, y referencias, estos documentos están la carpeta “Tienda de Mascotas”.

**Elegir un patrón de modelado:**

Patrón de modelado elegido: **Híbrido (Embedding + Referencing).**

**Justificación:**

- Embedding para datos que se consultan juntos y mejoran la lectura como items dentro de un pedido, notas dentro de mascota y detalles dentro de citas.
- Referencing para colecciones compartidas y que se actualizan a menudo o son muchas como cliente_id en pedidos, mascota_id en citas.

**RT:** Esto nos ayuda a realizar lecturas rápidas de documentos completos y con menor duplicación para datos compartidos también da equilibrio práctico para una tienda de mascotas.

**Crear el diagrama NoSQL del modelo:**