🎼 1. Como os dados são guardados numa base de dados SQLite na Google Drive?

O **SQLite** é um sistema de base de dados que guarda tudo num **único ficheiro** `.db`, que pode ser armazenado na Google Drive como qualquer outro ficheiro.

✅ Exemplo:

Se tiveres este esquema de tabela:

sql — Copiar — Editar

```sql
CREATE TABLE musicas (
    id INTEGER PRIMARY KEY,
    titulo TEXT,
    letra TEXT,
    imagem_path TEXT,
    data_escolha DATE
);
```

E adicionares isto:

sql — Copiar — Editar

```sql
INSERT INTO musicas (titulo, letra, imagem_path, data_escolha)
VALUES ('Senhor, tem piedade', 'Senhor, tem piedade de nós...', '/path/to/imagem1.jpg', '2025-07
```

O ficheiro `.db` guardará tudo isso internamente — textos, caminhos, datas etc.

> Podes sincronizar esse ficheiro `.db` com a Google Drive (com o app de desktop ou via scripts) para que esteja sempre atualizado.

🔍 2. IA para ver similaridade entre texto pesquisado e títulos na BD

Se quiseres que o utilizador pesquise por algo como `"piedade"` e o sistema sugira a música `"Senhor, tem piedade"`, podes usar **técnicas de NLP (Natural Language Processing)** para calcular **similaridade semântica ou textual**.

⚙️ Abordagens possíveis:

A. Similaridade com Embeddings (Recomendado)

Usa embeddings de frases para comparar vetores:

- Modelo: `all-MiniLM-L6-v2` do **SentenceTransformers**
- Biblioteca: `sentence-transformers`

python Copiar Editar

```python
from sentence_transformers import SentenceTransformer, util
import sqlite3

# carregar modelo
model = SentenceTransformer('all-MiniLM-L6-v2')

# texto do utilizador
query = "piedade"

# vetor da query
query_embedding = model.encode(query, convert_to_tensor=True)

# buscar dados da BD
conn = sqlite3.connect("musicas.db")
cursor = conn.cursor()
cursor.execute("SELECT id, titulo FROM musicas")
```

```python
# buscar dados da BD
conn = sqlite3.connect("musicas.db")
cursor = conn.cursor()
cursor.execute("SELECT id, titulo FROM musicas")
musicas = cursor.fetchall()

# calcular similaridade
results = []
for id_, titulo in musicas:
    titulo_embedding = model.encode(titulo, convert_to_tensor=True)
    score = util.cos_sim(query_embedding, titulo_embedding).item()
    results.append((id_, titulo, score))

# ordenar por relevância
results.sort(key=lambda x: x[2], reverse=True)

# mostrar top 3
for id_, titulo, score in results[:3]:
    print(f"{titulo} (similaridade: {score:.2f})")
```

Copiar Editar

B. Alternativa simples (fuzzy match textual)

Se quiseres algo mais leve:

python

Copiar Editar

```python
from fuzzywuzzy import fuzz
# ou rapidfuzz (mais rápido)

score = fuzz.partial_ratio("piedade", "Senhor, tem piedade")  # retorna valor de 0 a 100
```

🔄 3. Como fazer a ponte entre IA e a base de dados

Fluxo típico:

1. **Usuário escreve:** `"piedade"`
2. IA transforma esse texto em **vetor (embedding)**
3. Vai à base de dados buscar todos os títulos
4. IA calcula **similaridade** entre a pesquisa e os títulos
5. Mostra os títulos com maior semelhança

✅ Resumo

Questão	Resumo da Resposta
Como são guardados os dados?	Em ficheiro `.db` SQLite. Texto, caminhos e datas ficam todos lá, de forma local.
Como fazer busca inteligente?	Usa embeddings (ex: SentenceTransformers) ou fuzzy matching para medir similaridade.
Como integrar IA + BD?	Lê da BD → transforma títulos em vetores → calcula similaridade com a pesquisa.