
Panduan Implementasi RAG untuk Dokumen Terstruktur (CSV, Excel, JSON)

1. Latar Belakang

RAG (Retrieval-Augmented Generation) umumnya digunakan untuk mengelola dan mengambil informasi dari **dokumen tidak terstruktur**, seperti teks bebas dalam PDF, Word, atau HTML. Namun, dalam beberapa kasus, **dokumen terstruktur** seperti **CSV**, **Excel**, dan **JSON** juga perlu diproses agar informasi di dalamnya dapat dimanfaatkan oleh model LLM melalui mekanisme retrieval.

Dokumen terstruktur menyimpan informasi dalam bentuk tabel, kolom, atau objek, yang tidak bisa langsung digunakan oleh LLM tanpa diubah menjadi teks yang dapat direpresentasikan sebagai **embedding vector**.
Oleh karena itu, diperlukan langkah konversi ke dalam bentuk **chunk teks** sebelum data dimasukkan ke dalam **vector database**.

---

2. Prinsip Umum Pemrosesan Dokumen Terstruktur

Setiap dokumen terstruktur akan diubah menjadi kumpulan *chunk* teks.
Tujuannya agar setiap potongan data memiliki representasi semantik yang bisa dicari berdasarkan konteks pertanyaan pengguna.

- Setiap **chunk** akan menjadi satu entitas yang di-*embed* menjadi vektor.
- Informasi penting seperti nama kolom, nilai, dan konteks file akan dipertahankan agar retrieval tetap relevan.

---

3. Panduan Berdasarkan Format

A. JSON

Struktur yang Didukung

RAG hanya mendukung **JSON dalam bentuk list of objects**, misalnya:

```
[
  {"name": "Alice", "age": 30, "department": "HR"},
  {"name": "Bob", "age": 25, "department": "Engineering"}
]
```

Cara Pemrosesan

- Setiap **object** dalam list dianggap sebagai **satu chunk**.
- Konversikan setiap object menjadi teks deskriptif seperti:

Employee record:

- name: Alice

- age: 30

- department: HR

Atau dalam format satu kalimat:

Hasil Chunk

[

" Name: Alice, Age: 30, Department: HR",

" Name: Bob, Age: 25, Department: Engineering"

]

Setiap elemen di atas kemudian diubah menjadi embedding dan dimasukkan ke vector store.

B. CSV

Prinsip Pemrosesan

- Setiap **baris (row)** dalam tabel dianggap sebagai satu **chunk**.
- **Kolom (column)** berfungsi sebagai atribut atau konteks.
- Sistem akan membaca header kolom untuk membangun deskripsi setiap baris.

Contoh CSV

Name, Age, Department

Alice, 30, HR

Bob, 25, Engineering

Konversi ke Teks

Setiap baris diubah menjadi deskripsi teks seperti:

Row 1: Name: Alice, Age: 30, Department: HR.

Row 2: Name: Bob, Age: 25, Department: Engineering.

Hasil Chunk

[

" Name: Alice, Age: 30, Department: HR",

" Name: Bob, Age: 25, Department: Engineering"

]

C. Excel

Prinsip pemrosesan:

1. Setiap sheet akan diangaap instance yang terpisah
2. setiap sheet akan diconvert jadi markdown table
3. Setiap markdown akan di chunk dengan 5000 character/chunk sehingga informasi table akan terstore dengan lebih baik

loader excel hanya bisa load table, tidak bisa load informasi visual seperti gambar, chart dan plot.

---

8. Ringkasan

Format	**Chunk Level**	**Contoh Chunk**	**Catatan**
JSON	per object	{ "name": "Alice", "age": 30 }	List of object wajib
CSV	per row	Row 1: Name=Alice, Age=30	Header wajib
Excel	per table	|no | project 1| project2 | |1 | 432 | 435 |	table

---

Panduan Mengajukan Pertanyaan pada Dokumen Terstruktur

Sistem ini mendukung pencarian dan pemahaman terhadap **dokumen terstruktur** seperti **CSV**, **Excel**, dan **JSON (list of objects)**.
Setiap **baris** (pada CSV/Excel) atau **object** (pada JSON) dianggap sebagai **satu unit data atau chunk** yang diproses secara terpisah dan disimpan ke dalam sistem pencarian berbasis vektor.

Karena itu, **pertanyaan yang Anda ajukan sebaiknya berfokus pada data per item**, bukan keseluruhan dokumen.
Sistem tidak melakukan pemindaian penuh terhadap seluruh isi file, tetapi hanya pada sejumlah chunk yang paling relevan dengan pertanyaan Anda.

Pertanyaan yang Dapat Diajukan

Anda dapat bertanya tentang **detail atau informasi spesifik dari satu entitas** yang ada di dalam dokumen.

Contoh:

- “Siapa penanggung jawab proyek Huluantu?”
- “Berapa estimasi biaya untuk proyek Huluantu?”
- “Kapan proyek Huluantu dimulai?”
- “Apa status proyek Huluantu saat ini?”
- “Proyek Huluantu dikerjakan oleh siapa?”
- “Apa tujuan proyek Huluantu?”
- “Apakah proyek Huluantu sudah selesai?”

Dengan kata lain, pastikan pertanyaan Anda **menyebutkan nama atau identitas objek tertentu**, seperti nama proyek, nama orang, nama perusahaan, dan sebagainya.

Pertanyaan yang Tidak Dapat Diajukan (atau Terbatas)

Pertanyaan yang membutuhkan **pemahaman menyeluruh terhadap seluruh dokumen** tidak dapat dijawab secara akurat.
Sistem hanya mengambil sebagian kecil data (misalnya 10 atau 20 baris/object paling relevan), sehingga hasilnya tidak mewakili keseluruhan isi dokumen.

Contoh pertanyaan yang tidak dapat dijawab secara lengkap:

- “Sekarang ada proyek apa saja?”
- “Proyek apa saja yang dimulai pada tahun 2023?”
- “Berapa total proyek yang sedang berjalan?”
- “Berapa total anggaran dari semua proyek?”
- “Proyek mana yang memiliki nilai kontrak tertinggi?”
- “Berapa banyak proyek yang sudah selesai?”
- “Tampilkan semua proyek di Kalimantan.”

Pertanyaan semacam ini memerlukan pemrosesan seluruh data dalam dokumen, sedangkan sistem ini bekerja berdasarkan potongan data yang terbatas jumlahnya.

Tips

- Gunakan pertanyaan yang **spesifik dan kontekstual**, seperti menanyakan informasi tentang satu proyek, satu orang, atau satu entitas tertentu.
- Hindari pertanyaan yang bersifat **global atau agregatif**, seperti “semua”, “total”, “daftar lengkap”, atau “keseluruhan”.
- Jika Anda ingin mencari entitas tertentu, sertakan **nama atau kata kunci unik** yang ada di data (misalnya nama proyek, perusahaan, atau wilayah).

Dengan mengikuti panduan ini, sistem akan dapat memberikan jawaban yang lebih relevan, akurat, dan sesuai dengan data terstruktur yang telah Anda unggah.