NO SQL database

Let's contrast the structure of the relational database with a hypothetical NoSQL database, focusing on MongoDB as an example. (Note: The transition to NoSQL often involves denormalization and a different way of thinking about data modeling)

Aspect	Relational Database (PostgreSQL)	NoSQL Database (MongoDB)
Schema Flexibility	Fixed schema with predefined tables and relationships.	Flexible schema allowing dynamic and evolving data structures.
Normalization	Normalized for reduced redundancy. JOIN operations for queries.	Denormalized for performance, allowing data duplication for speed.
Query Language	SQL for structured queries.	MongoDB Query Language (MQL) for JSON-like documents.
Horizontal Scaling	Typically scales vertically (adding resources to a single server)	Scales horizontally (adding servers to a distributed system).
Use Case	Well-suited for complex transactions and structured data.	Ideal for large volumes of unstructured or semi-structured data.
Atomic Transactions	Supports ACID transactions.	Sacrifices full ACID transactions for better scalability.

Below is a sample of how the **Patient Record Collection** data will look like if it is maintained in a NOSQL database like mongoDB.


```
{
  "patient_record": [
    {
      "patient_id": 1,
      "patient_name": "John Doe",
      "patient_age":30,
      "patient_gender" : "Male",
      "city_name":"New York City",
      "state_name": "New York",
      "country_name": "USA",
      "diagnoses": [
        {
          "diagnosis_id": 1,
          "diagnosis_date": "2023-11-01",
          "doctor": [{"doctor_id" : 5, "doctor_name" : "Dr.Taylor", "doctor_speciality": "Cardiologist",
"experience_years":11}],
          "disease": [{"disease_id" : 10, "disease_name": "Angina Pectoris", "disease_description" : "Chest pain or
discomfort caused by reduced blood flow to the heart muscle"}],
          "disease_severity" : [{"disease_severity_code" : 3, "disease_severity_description" : "Severe"}],
          "weight_kg": 70.50,
          "temperature_f": 98.60,
          "systolic_blood_pressure_mmhg": 120.00,
          "diastolic_blood_pressure_mmhg": 80.00
        },
        {
          "diagnosis_id": 46,
          "diagnosis_date": "2023-11-10",
```

```
        "doctor": [{"doctor_id" : 5, "doctor_name" : "Dr.Taylor", "doctor_speciality": "Cardiologist",
"experience_years":11}],
        "disease": [{"disease_id" : 10, "disease_name": "Angina Pectoris", "disease_description" : "Chest pain or
discomfort caused by reduced blood flow to the heart muscle"}],
        "disease_severity" : [{"disease_severity_code" : 2, "disease_severity_description" : "Moderate"}],
        "weight_kg": 72.00,
        "temperature_f": 99.00,
        "systolic_blood_pressure_mmhg": 122.00,
        "diastolic_blood_pressure_mmhg": 82.00
       },
       {
        "diagnosis_id": 92,
        "diagnosis_date": "2023-11-20",
        "doctor": [{"doctor_id" : 5, "doctor_name" : "Dr.Taylor", "doctor_speciality": "Cardiologist",
"experience_years":11}],
        "disease": [{"disease_id" : 10, "disease_name": "Angina Pectoris", "disease_description" : "Chest pain or
discomfort caused by reduced blood flow to the heart muscle"}],
        "disease_severity" : [{"disease_severity_code" : 1, "disease_severity_description" : "Mild"}],
        "weight_kg": 75.50,
        "temperature_f": 98.50,
        "systolic_blood_pressure_mmhg": 125.00,
        "diastolic_blood_pressure_mmhg": 85.00
       }
     ]
   }
 ]
}
```


Explanation:

- This structure captures information about patients, including their personal details and a history of diagnoses.
- Each diagnosis includes details about the doctor who conducted it, the diagnosed disease, its severity, and relevant measurements.
- The structure is hierarchical, providing a clear representation of the relationships between patients, diagnoses, doctors, diseases, and severity levels.

Advantages Over Relational Database:

- **Nested Relationships:** The hierarchical and nested structure of the JSON data allows for the representation of complex relationships, such as the association between patients, diagnoses, doctors, diseases, and severity levels. This can be more intuitive and natural in a NoSQL setting.
- **Flexibility and Schema-less Design:** The flexibility of MongoDB allows each patient document to have a different number of diagnoses or additional fields as needed.
- **Atomic Updates:** MongoDB supports atomic updates at the document level. We can update specific fields within a diagnosis without affecting the entire patient document.
- **Denormalization:** Information about a patient's diagnoses is denormalized by storing it directly within the patient document, avoiding the need for complex joins.
- **Embedding Data:** The "diagnoses" array is embedded within each patient document, facilitating retrieval of a patient's entire medical history with a single query.
- **Query Performance:** The document-oriented model can enhance read performance, especially when accessing a patient's diagnosis history, as all relevant data is stored together.