```
db.movies.insert({
 movie_id: "The Godfather",
 actors: [
  { id: "Amit Garg", sex: "M" },
  { id: " Another Actor", sex: "F"}
 ],
 directors: [
  { id: "FrancisCoppola001" sex: "M" }
 ],
 genres: ["Suspense", "Drama"],
 ratings: [
  { rank: 5, votes: 1000 },
  { rank: 10, votes: 500 },
  { rank: 5, votes: 300 }
 ]
})
```

No processo de transformação para NoSQL, eu organizei o objeto **filme** de forma que todas as informações relacionadas ficassem encapsuladas dentro de um único documento JSON. Ao invés de armazenar dados de atores, diretores, gêneros e avaliações em tabelas separadas e fazer referências cruzadas entre elas, movi essas informações diretamente para o próprio documento do filme.

Atores e diretores, por exemplo, são armazenados em listas, como actors e directors, dentro do documento, e as avaliações ficam em uma lista chamada ratings.

Essa abordagem, conhecida como desnormalização, elimina a necessidade de manter tabelas associativas e simplifica o acesso às informações. Ao utilizar subdocumentos para representar atores e diretores, contendo apenas os campos essenciais como id e sex, reduzi a complexidade, facilitando a consulta dos dados sem depender de múltiplos joins.

Com essa estrutura, todas as informações relevantes para um filme estão diretamente disponíveis, o que melhora o desempenho de leitura, já que não é preciso buscar dados em várias tabelas.

Isso simplifica o processo de consulta e manipulação dos dados, tornando-o mais eficiente, especialmente em sistemas que não precisam de atualizações frequentes.