1. Nested İlişkileri

Nested ilişkiler, bir dokümanın içinde başka bir dokümanın yer aldığı senaryolar için kullanılır. Örneğin, bir kullanıcı ve onun yorumları arasında bir nested ilişki kurabiliriz.

a. Veri Yapısı

Örneğin, bir `user` dokümanı ve ona bağlı `comments` (yorumlar) nested alanı oluşturalım.

```json
PUT /users
{
  "mappings": {
    "properties": {
      "name": { "type": "text" },
      "comments": {
        "type": "nested",
        "properties": {
          "text": { "type": "text" },
          "date": { "type": "date" }
        }
      }
    }
  }
}
```

Bu yapı, her bir kullanıcı dokümanının içinde birden fazla yorum barındırmasını sağlar.

b. Veri Ekleme

Kullanıcı verilerini ekleyelim:

```json
POST /users/_doc/1
{
  "name": "Alice",
  "comments": [
    { "text": "Great product!", "date": "2023-10-01" },
    { "text": "Could be better.", "date": "2023-10-02" }
  ]
}

POST /users/_doc/2
{
  "name": "Bob",
  "comments": [
```

```
    { "text": "I love it!", "date": "2023-10-01" }
  ]
}
```

c. Nested Sorgu

Kullanıcılardan sadece belirli bir yorumu içerenleri sorgulamak için `nested` sorgusu kullanabilirsiniz. Örneğin, "Great product!" yorumunu içeren kullanıcıları bulmak:

```json
GET /users/_search
{
  "query": {
    "nested": {
      "path": "comments",
      "query": {
        "match": { "comments.text": "Great product!" }
      }
    }
  }
}
```

Bu sorgu, `comments` alanında "Great product!" içeren kullanıcıları döndürecektir.

2. Parent-Child İlişkileri

Parent-child ilişkileri, bir dokümanın (parent) başka bir dokümanı (child) referans aldığı durumlar için kullanılır. Bu ilişki, veri modelinin karmaşıklaştığı senaryolarda esneklik sağlar.

a. Veri Yapısı

Bir `article` ve bu makaleye ait `comments` (yorumlar) arasında bir parent-child ilişkisi kuralım:

```json
PUT /articles
{
  "mappings": {
    "properties": {
      "title": { "type": "text" },
      "content": { "type": "text" },
      "comments": {
        "type": "join",
        "relations": {
          "article": "comment"
        }
      }
```

```
    }
  }
}
```

Burada `join` tipi, parent-child ilişkisini oluşturur.

b. Veri Ekleme

Önce bir makale ekleyelim, ardından ona bağlı yorumlar ekleyelim:

```json
POST /articles/_doc/1?refresh
{
  "title": "Elasticsearch Basics",
  "content": "This article explains the basics of Elasticsearch.",
  "comments": "article"
}

POST /articles/_doc/2?routing=1
{
  "text": "Great article!",
  "comments": "comment"
}

POST /articles/_doc/3?routing=1
{
  "text": "Very informative.",
  "comments": "comment"
}
```

Burada, `routing` parametresi, child dokümanların doğru parent ile ilişkilendirilmesi için kullanılır.

c. Parent-Child Sorgusu

Bir makaleye ait yorumları sorgulamak için aşağıdaki gibi bir sorgu yapabilirsiniz:

```json
GET /articles/_search
{
  "query": {
    "has_child": {
      "child_type": "comment",
      "query": {
        "match": { "text": "Great article!" }
      }
    }
  }
```

}
```

Bu sorgu, "Great article!" yorumu olan makaleleri döndürecektir.

### 3. Hangi Durumda Hangisi Kullanılmalı?

- **Nested İlişkiler**: Eğer dokümanlar arasında sıkı bir ilişki varsa ve aynı dokümanda birden fazla kayıt gerekiyorsa kullanın. Yorumlar, kullanıcılarla sıkı bir ilişki içindedir.

- **Parent-Child İlişkileri**: Eğer dokümanlar arasında daha gevşek bir ilişki varsa ve çocuk dokümanların sayısı değişkense kullanın. Örneğin, makale ve yorumlar arasında.
```