MongoDB

A DOCUMENT-ORIENTED DATABASE

Indici

Indici

La costruzione di indici consente di migliorare le performance in lettura

- ATTENZIONE: gli indici vengono aggiornati ad ogni modifica; il rischio è di peggiorare le performance in scrittura

Un esempio: db.products.createIndex({category: 1, price: -1})

- :1 indica un ordinamento crescente, -1 decrescente
- L'ordine dei campi è importante

Caratteristiche

- Si possono indicizzare gli array
- Esistono tipi speciali di indici: geospaziali, di testo, hash
- Indici parziali: indicizzano solo alcuni documenti sulla base di una query
- Indici sparsi: indicizzano solo i documenti per cui il campo non esiste
- Indici TTL: cancellano il documento se la data indicizzata è obsoleta

Indici di testo

Consentono di fare ricerche di testo all'interno di uno o più campi

- ATTENZIONE: può essere creato un solo indice di testo per collezione
- db.stores.createIndex({ name: "text", description: "text" })

L'indice esclude punteggiatura e *stop-words* e riduce i termini a radice

- Esempio: "Collezione con indice di testo." → ["collezion", "indic", "test"]

Caratteristiche

- Si possono indicizzare campi testuali e array di stringhe
- Si possono assegnare pesi diversi a campi diversi
- Tarato di default sull'inglese, ma si possono impostare lingue diverse

Operatore $text

```
db.stores.insert([
  { _id: 1, name: "Java Hut", description: "Coffee and cakes" },
  { _id: 2, name: "Burger Buns", description: "Gourmet hamburgers" },
  { _id: 3, name: "Coffee Shop", description: "Just coffee" },
  { _id: 4, name: "Clothes Clothes", description: "Discount clothing" },
  { _id: 5, name: "Java Shopping", description: "Indonesian goods" }
])

db.stores.createIndex( { name: "text", description: "text" } )
```

Operatore $text

```
db.stores.insert([
  { _id: 1, name: "Java Hut", description: "Coffee and cakes" },
  { _id: 2, name: "Burger Buns", description: "Gourmet hamburgers" },
  { _id: 3, name: "Coffee Shop", description: "Just coffee" },
  { _id: 4, name: "Clothes Clothes", description: "Discount clothing" },
  { _id: 5, name: "Java Shopping", description: "Indonesian goods" }
])
```

```
db.stores.find( { $text: { $search: "java coffee shop" } } )
```

- Le parole indicate sono considerate in OR

Operatore $text

```
db.stores.insert([
  { _id: 1, name: "Java Hut", description: "Coffee and cakes" },
  { _id: 2, name: "Burger Buns", description: "Gourmet hamburgers" },
  { _id: 3, name: "Coffee Shop", description: "Just coffee" },
  { _id: 4, name: "Clothes Clothes", description: "Discount clothing" },
  { _id: 5, name: "Java Shopping", description: "Indonesian goods" }
])
```

```
db.stores.find( { $text: { $search: "java \"coffee shop\"" } } )
```

- Le stringhe indicate tra virgolette (") vengono cercate esattamente

Operatore $text

```
db.stores.insert([
  { _id: 1, name: "Java Hut", description: "Coffee and cakes" },
  { _id: 2, name: "Burger Buns", description: "Gourmet hamburgers" },
  { _id: 3, name: "Coffee Shop", description: "Just coffee" },
  { _id: 4, name: "Clothes Clothes", description: "Discount clothing" },
  { _id: 5, name: "Java Shopping", description: "Indonesian goods" }
])
```

```
db.stores.find( { $text: { $search: "java shop -coffee" } } )
```

- Le parole precedute dal meno (-) causano l'esclusione del documento

Operatore $text

```
db.stores.insert([
  { _id: 1, name: "Java Hut", description: "Coffee and cakes" },
  { _id: 2, name: "Burger Buns", description: "Gourmet hamburgers" },
  { _id: 3, name: "Coffee Shop", description: "Just coffee" },
  { _id: 4, name: "Clothes Clothes", description: "Discount clothing" },
  { _id: 5, name: "Java Shopping", description: "Indonesian goods" }
])
```


```
db.stores.find(
  { $text: { $search: "java coffee shop" } },
  { score: { $meta: "textScore" } }
).sort( { score: { $meta: "textScore" } } )
```

- E'possibile ordinare i risultati sulla base di un punteggio di similarità con la query di ricerca