ITS TECNOLOGIE DELL'INFORMAZIONE E DELLA COMUNICAZIONE

NOME CORSO

Fintech Software Developer

Unità Formativa (UF): Basi di dati SQL

Docente: Durando Giulio

Titolo argomento: Gli indici in MongoDB

Gli indici in MongoDB

Gli indici MongoDB sono un modo eccellente per ottimizzare e organizzare i database e sono relativamente semplici da impostare.

Come con qualsiasi database, gli indici sono strutture di dati essenziali in MongoDB. Favoriscono semplici query sul database e consentono di archiviare e organizzare una parte della collection in campi univoci. È possibile sfruttare la potenza degli indici per rendere le query più veloci ed efficienti.

Gli indici in MongoDB consentono di dare la priorità ad alcuni campi in un documento e di trasformarli in parametri di query in un secondo momento. Quando si crea una raccolta, MongoDB crea un indice ID predefinito. Ma è possibile aggiungerne altri se si hanno domande o esigenze di accordo più elevate.

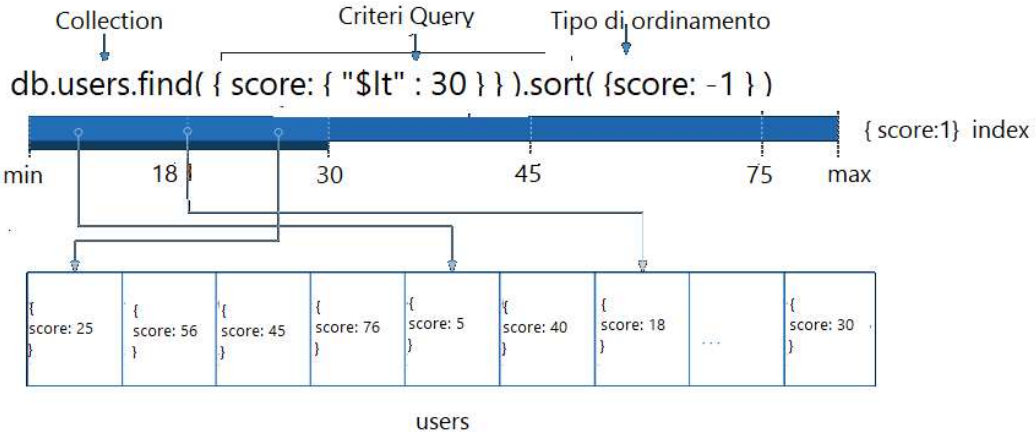
Gli indici univoci prevengono anche i duplicati durante l'immissione dei dati. Sono utili per rifiutare voci che già esistono nel database.

Gli indici aiutano a trovare in modo specifico ciò che si desidera utilizzando il formato di query definito nella quantità richiesta senza scansionare un'intera raccolta. Quando si crea un indice, è possibile specificare come si desidera che i dati vengano ordinati ogni volta che lo si richiede.

Gli indici supportano l'esecuzione efficiente delle query in MongoDB. Senza indici, MongoDB deve eseguire una *scansione della collection* , ovvero scansionare ogni documento in una raccolta, per selezionare quei documenti che corrispondono all'istruzione della query. Se esiste un indice appropriato per una query, MongoDB può utilizzare l'indice per limitare il numero di documenti che deve ispezionare.

Gli indici sono strutture di dati speciali che memorizzano una piccola parte del set di dati della raccolta in una forma facile da attraversare. L'indice memorizza il valore di un campo specifico o di un insieme di campi, ordinato in base al valore del campo. L'ordinamento delle voci dell'indice supporta efficienti corrispondenze di uguaglianza e operazioni di query basate su intervalli. Inoltre, MongoDB può restituire risultati ordinati utilizzando l'ordinamento nell'indice.

Il diagramma seguente illustra una query che seleziona e ordina i documenti corrispondenti utilizzando un indice:



Fondamentalmente, gli indici in MongoDB sono simili agli indici in altri sistemi di database. MongoDB definisce gli indici a livello di collection e supporta gli indici su qualsiasi campo o sottocampo dei documenti in una raccolta MongoDB.

MongoDB fornisce un supporto completo per gli indici su qualsiasi campo in una collection di documenti. Per impostazione predefinita, tutte le collections hanno un indice sul campo **_id** e le applicazioni e gli utenti possono aggiungere ulteriori indici per supportare query e operazioni importanti.

Per creare un indice in una collection in mongoshell , si deve usare il comando:

```
db.collection.createIndex( <key and index type specification>, <options> )
```

L'esempio seguente crea un indice con ordine decrescente a chiave singola nel campo name:

```
db.collection.createIndex( { name: -1 } )
```

Il metodo **db.collection.createIndex()** crea un indice solo se non esiste già un indice con la stessa specifica.

Nomi degli indici

Il nome predefinito per un indice è la concatenazione delle chiavi indicizzate e la direzione di ciascuna chiave nell'indice (ad esempio 1 o -1) utilizzando i caratteri di sottolineatura come separatore. Ad esempio, un indice creato su **{ item : 1 }** ha nome **item_1**, un indice creato su **{ item : 1, quantity: -1** } ha nome **item_1_quantity_-1**.

È possibile creare indici con un nome personalizzato, ad esempio uno più leggibile rispetto a quello predefinito. Si consideri, ad esempio, un'applicazione che interroga frequentemente la collection products per popolare i dati nell'inventario esistente. Il metodo **createIndex()** seguente crea un indice su item e quantity denominato query for inventory:

```
db.products.createIndex(
  { item: 1, quantity: -1 } ,
  { name: "query for inventory" }
)
```

È possibile visualizzare i nomi degli indici utilizzando il metodo **db.collection.getIndexes()**. Non è possibile rinominare un indice una volta creato. È necessario eliminare e ricreare l'indice con un nuovo nome.

Tipi di indici

Indici a campo singolo

Oltre all'indice **_id** predefinito da MongoDB, MongoDB supporta la creazione di indici crescenti/decrescenti definiti dall'utente su un singolo campo di un documento.

Per un indice a campo singolo e operazioni di ordinamento, l'ordinamento (ad es. crescente o decrescente) della chiave dell'indice non ha importanza poiché MongoDB può usare l'indice in entrambe le direzioni.

Consideriamo una collection denominata records che contiene documenti simili al seguente documento di esempio:

```
{
  "_id": ObjectId("570c04a4ad233577f97dc459"),
  "score": 1034,
  "location": { state: "NY", city: "New York" }
}
```

L'operazione seguente crea un indice crescente sul campo score della collection records:

```
db.records.createIndex( { score: 1 } )
```

Il valore del campo nella specifica dell'indice descrive il tipo di indice per quel campo. Ad esempio, un valore di `1` specifica un indice che ordina gli articoli in ordine crescente. Un valore di `-1` specifica un indice che ordina gli elementi in ordine decrescente.

L'indice creato supporterà le query che selezionano ill campo score, come le seguenti:

```
db.records.find( { score: 2 } )
db.records.find( { score: { $gt: 10 } } )
```

È possibile creare **indici sui campi all'interno di documenti incorporati**, proprio come è possibile indicizzare i campi di primo livello nei documenti. Gli indici sui campi incorporati differiscono dagli indici sui documenti incorporati, che includono l'intero contenuto fino alla massima dimensione dell'indice del documento incorporato nell'indice. Al contrario, gli indici sui campi incorporati consentono di utilizzare una "notazione a punti" per introspezione nei documenti incorporati.

Consideriamo sempre la collection denominata records vista precedentemente. L'operazione seguente crea un indice sul campo location.state:

```
db.records.createIndex( { "location.state": 1 } )
```

L'indice creato supporterà le query che selezionano il campo `location.state`, come le seguenti:

```
db.records.find( { "location.state": "CA" } )
db.records.find( { "location.city": "Albany", "location.state": "NY" } )
```

È possibile anche creare **indici sul documento incorporato nel suo insieme**.

Consideriamo sempre la collection records. Il campo location è un documento incorporato, contenente i campi incorporati city e state. Il comando seguente crea un indice sul campo location nel suo insieme:

```
db.records.createIndex( { location: 1 } )
```

La seguente query può utilizzare l'indice sul campo location:

```
db.records.find( { location: { city: "New York", state: "NY" } } )
```

Sebbene la query possa utilizzare l'indice, il set di risultati fornito da questa query non include il documento di esempio precedente. Quando si eseguono corrispondenze di uguaglianza sui documenti incorporati, l'ordine dei campi nei documenti incorporati devono corrispondere esattamente e in questo caso l'ordine dei due campi non è uguale a quello impostato nella query.

Indici a campo Composto

MongoDB supporta anche indici definiti dall'utente su più campi. L'ordine dei campi elencati in un indice composto ha significato. Ad esempio, se un indice composto è costituito da **{userid: 1, score: -1}**, l'indice ordina prima per userid in ordine crescente e poi, all'interno di ogni valore di userid, ordina per score in ordine decrescente.

Per creare un indice composto si deve utilizzare un comando simile alla seguente prototipo

```
db.collection.createIndex( { <field1>: <type>, <field2>: <type2>, ... } )
```

L'ordine dei campi indicizzati ha un forte impatto sull'efficacia di un particolare indice per una determinata query. Per la maggior parte degli indici composti, se si segue la regola ESR aiuta a creare indici efficienti.

L'operazione seguente crea un indice crescente nei campi item e stock della collection products

```
db.products.createIndex( { "item": 1, "stock": 1 } )
```

L'ordine dei campi elencati in un indice composto è importante. L'indice conterrà i riferimenti ai documenti ordinati prima in base ai valori del campo item e, all'interno di ciascun valore del campo item, ordinati in base ai valori del campo stock.

Oltre a supportare le query che corrispondono a tutti i campi dell'indice, gli indici composti possono supportare le query che corrispondono un sottoinsieme dei campi dell'indice. Cioè, l'indice supporta sia le query sul campo item sia quelle su entrambi i campi item e stock:

```
db.products.find( { item: "Banana" } )
db.products.find( { item: "Banana", stock: { $gt: 5 } } )
```

Prefissi

I prefissi dell'indice sono i sottoinsiemi *iniziali* dei campi indicizzati. Si consideri ad esempio il seguente indice composto:

```
{ "item": 1, "location": 1, "stock": 1 }
```

L'indice ha i seguenti prefissi di indice:

- **{ item: 1 }**
- **{ item: 1, location: 1 }**

Con un indice composto, MongoDB può utilizzare l'indice per supportare le query sui prefissi dell'indice. Pertanto, MongoDB può utilizzare l'indice per le query sui seguenti campi:

- Il campo item,
- Il campo item e il campo location,
- Il campo item, il campo location *e* il campo stock.

Intersezione Indici

MongoDB può utilizzare l'intersezione di più indici per soddisfare le query. In generale, ogni intersezione di indici coinvolge due indici; tuttavia, MongoDB può utilizzare intersezioni di indici multiple/nidificate per risolvere una query.

Per spiegare l'intersezione degli indici, consideriamo una collection orders con i seguenti indici:

```
{ qty: 1 }
{ item: 1 }
```

MongoDB può utilizzare l'intersezione dei due indici per supportare la seguente query:

```
db.orders.find( { item: "abc123", qty: { $gt: 15 } } )
```

Indici TTL

Gli indici TTL sono speciali indici a campo singolo che MongoDB può utilizzare per rimuovere automaticamente i documenti da una raccolta dopo un certo periodo di tempo o a un'ora specifica. La scadenza dei dati è utile per alcuni tipi di informazioni come dati di eventi generati dalla macchina, registri e informazioni sulla sessione che devono persistere in un database solo per un periodo di tempo limitato.

Per creare un indice TTL, utilizzare il metodo **createIndex()** su un campo il cui valore è una **Data** o un array che contiene valori di data e specificare l'opzione **expireAfterSeconds** con il valore TTL desiderato in secondi.

Ad esempio, per creare un indice TTL sul campo **lastModifiedDate** della collection **eventlog**, con un valore TTL di **3600** secondi, utilizzare la seguente operazione in mongosh

```
db.eventlog.createIndex( { "lastModifiedDate": 1 }, { expireAfterSeconds: 3600 } )
```

Gestione degli indici

Visualizzazione degli indici esistenti

Per visualizzare l'elenco di tutti gli indici di una collection, utilizzare il metodo

```
db.collection.getIndexes()
```

Rimozione degli indici

Quando si rimuovono gli indici nella shell MongoDB, è possibile:

- Rimuovere un indice specifico.
- Rimuovere tutti gli indici dalla raccolta.

Per **rimuovere un indice specifico** si deve utilizzare il metodo

```
db.collection.dropIndex()
```

Ad esempio, l'operazione seguente rimuove un indice crescente sul campo **tax-id** nella collection **accounts**

```
db.accounts.dropIndex( { "tax-id": 1 } )
```

L'operazione restituisce un documento con lo stato dell'operazione:

```
{ "nIndexesWas" : 3, "ok" : 1 }
```

Dove il valore di **nIndexesWas** riflette il numero di indici *prima* di rimuovere questo indice.

Per **rimuovere tutti gli indici** di una collection, tranne l'indice _id, si deve utilizzare il metodo

```
db.collection.dropIndexes()
```

Ad esempio, il comando seguente rimuove tutti gli indici dalla collection accounts:

```
db.accounts.dropIndexes()
```

Modifica un indice

Per modificare un indice esistente, tranne gli indici TTL, nella MongoDB Shell, è necessario eliminare e ricreare l'indice.