ITS TECNOLOGIE DELL'INFORMAZIONE E DELLA COMUNICAZIONE

NOME CORSO

Fintech Software Developer

Unità Formativa (UF): Basi di dati SQL

Docente: Durando Giulio

Titolo argomento: Operazioni CRUD

Operazioni CRUD

Le operazioni CRUD (**C**reate, **R**ead, **U**pdate, **D**elete) permettono di creare, leggere, aggiornare ed eliminare documenti all'interno di un database.

Possono essere eseguite sia tramite l'interfaccia a linea di comando che tramite una qualsiasi interfaccia grafica (Compass, 3TStudio, ecc). La sintassi che si deve utilizzare è sempre quella della notazione JSON (gli oggetti iniziano con la parentesi graffa aperta e terminano con la parentesi graffa chiusa, le matrici iniziano con la parentesi quadra aperta e terminano con la parentesi quadra chiusa, i campi vanno separati da una virgola)

Create

Le operazioni di creazione o inserimento aggiungono nuovi documenti a una collection. Se la raccolta non esiste, le operazioni di creazione creano anche la raccolta. È possibile inserire un singolo documento o più documenti in un'unica operazione.

Read

Le operazioni di lettura recuperano documenti da una collection; cioè interrogano una collection per i documenti. È possibile specificare criteri o filtri che identificano i documenti da restituire.

Update

Le operazioni di aggiornamento modificano i documenti esistenti in una collection. È possibile aggiornare un singolo documento o più documenti in un'unica operazione. È possibile specificare criteri o filtri che identificano i documenti da aggiornare. Questi filtri utilizzano la stessa sintassi delle operazioni di lettura.

Delete

Le operazioni di eliminazione rimuovono i documenti esistenti da una collection. È possibile rimuovere un singolo documento o più documenti in un'unica operazione. È possibile specificare criteri o filtri che identificano i documenti da rimuovere. Questi filtri utilizzano la stessa sintassi delle operazioni di lettura.

Operazioni di Create (Inserimento)

La shell MongoDB fornisce i seguenti metodi per inserire documenti in una collection:

- Per inserire un singolo documento, utilizzare **db.collection.insertOne()**
- Per inserire più documenti, utilizzare **db.collection.insertMany()**
- Per inserire uno o più documenti, utilizzare **db.collection.insert()** (sconsigliato)

In MongoDB, ogni documento archiviato in una collection richiede un campo _id univoco che funge da chiave primaria. Se nell'inserimento di un documento di omette il campo _id, il driver MongoDB genera automaticamente un ObjectId per il campo _id.

db.collection.insertOne()

Inserisce un *singolo* documento in una collection. Se il documento non specifica un campo _id, MongoDB aggiunge il campo _id con un valore ObjectId al nuovo documento

Per inserire un nuovo documento nella **collection movies** del **database archivio_film** si deve usare la seguente sintassi:

```
use archivio_film

db.movies.insertOne(
  {
    titolo: "Top Gun Maverick",
    genere: [ "Azione", "Avventura", “Drammatico” ],
    durata: 131,
    anno: 2022,
    classificazione: “+13”,
    regista: "Joseph Kosinski",
    cast: [ "Tom Cruise", "Jennifer Connelly", "Val Kilmer" ]
  }
)
```

insertOne() inserisce un documento che include il valore del campo _id del documento appena inserito.

db.collection.insertMany()

Inserisce *più* documenti in una collection. È necessario passare una matrice di documenti al metodo. Se i documenti non specificano un campo _id, MongoDB aggiunge il campo _id con un valore ObjectId a ciascun documento

Per inserire due nuovi documenti nella **collection movies** del **database archivio_film** si deve usare la seguente sintassi:

```
use archivio_film

db.movies.insertMany()
(
   [
        {
           titolo: "Top Gun Maverick",
           genere: [ "Azione", "Avventura", “Drammatico” ],
           durata: 131,
           anno: 2022,
           classificazione: PG”,
           regista: "Joseph Kosinski",
           cast: [ "Tom Cruise", "Jennifer Connelly", "Val Kilmer" ]
        },
        {
           titolo: "Elvis",
           genere: [ "Biografico", "Musicale", “Drammatico” ],
           durata: 157,
           anno: 2022,
           classificazione: “PG-13”,
           regista: "Baz Luhrmann",
           cast: [ "Austin Butler", "Tom Hanks", "Olivia De Jonge" ]
        }
   ]
)
```

Operazioni di Read (lettura)

La shell MongoDB fornisce il seguente metodo per visualizzare documenti in una collection:

- **db,collection.Find()**

db,collection.find()

Il metodo **find()** accetta in ingresso due parametri, entrambi opzionali: i criteri di filtraggio e le specifiche di proiezione.

Tramite il primo si stabilisce quali documenti cercare, con il secondo invece quali campi dei documenti trovati devono o non devono essere restituiti. Per essere precisi, **find()** non restituisce un elenco di documenti ma un cursore, ossia un puntatore ad un elenco che deve essere iterato per ottenere i documenti veri e propri. La console di mongo si occupa per noi di iterare il cursore e mostrare a video i risultati. Usando i driver, invece, è responsabilità nostra iterare il cursore nel modo specifico per il driver ed il linguaggio usato.

Criteri di filtraggio

Non specificare i criteri di filtraggio comporta la restituzione di tutti i documenti (nessun filtraggio), mentre non specificando la proiezione si ottengono i documenti interi. Quindi, per ottenere tutti i documenti (con tutti i relativi campi) di una collection, possiamo invocare la funzione senza parametri:

Il seguente codice restituisce tutti i documenti della collection movies

```
use archivio_film
db.moovies.find()
```

Questa operazione è equivalente alla seguente istruzione SQL in un database SQL

*SELECT * FROM movies*

Specificare condizioni di ricerca

Per restituire tutti i film con **titolo uguale a "Elvis"** della collection movies si deve dare il seguente comando

```
use archivio_film
db.movies.find( { titolo: "Elvis" } )
```

Questa operazione è equivalente alla seguente istruzione SQL

*SELECT * FROM movies WHERE titolo = "Elvis"*

Se il campo per cui viene fatta la ricerca è un array, verrà cercata una corrispondenza all'interno dell'array. Ad esempio, il seguente codice:

```
use archivio_film
db.movies.find( { cast: "Tom Hanks" } )
```

restituirà tutti i documenti in cui all'interno dell'array relativa al cast comparirà "Tom Hanks"

La ricerca può essere fatta anche all'interno dei documenti incorporati.

Operatori di query

Un altro tipo di filtraggio applicabile utilizza gli operatori forniti da MongoDB per la ricerca che sono quelli della seguente tabella

Nome	Descrizione
$eq	Corrisponde a valori uguali a un valore specificato.
$gt	Corrisponde a valori maggiori di un valore specificato.
$gte	Corrisponde a valori maggiori o uguali a un valore specificato.
$in	Corrisponde a uno qualsiasi dei valori specificati in una matrice.
$lt	Corrisponde a valori inferiori a un valore specificato.
$lte	Corrisponde a valori minori o uguali a un valore specificato.
$ne	Corrisponde a tutti i valori che non sono uguali a un valore specificato.
$nin	Non corrisponde a nessuno dei valori specificati in una matrice.

È possibile utilizzare gli operatori di query in un documento di filtro di query per eseguire confronti e valutazioni più complessi. Gli operatori di query in un documento di filtro di query hanno la forma seguente:

{ <field1>: { <operator1>: <value1> }, ... }

Esempio

Il seguente codice trova tutti i documenti in cui la durata è maggiore o uguale a 120 minuti

```
db.movies.find( { durata: { $gte: 120 } } )
```

Il seguente codice restituisce tutti i film della collection movies del database archivio_film classificati PG o PG-13

```
use archivio_film
db.movies.find( { classificazione: { $in: [ "PG", "PG-13" ] } } )
```

Questa operazione corrisponde alla seguente istruzione SQL:

*SELECT * FROM movies WHERE classificazione in ("PG", "PG-13")*

È possibile effettuare la selezione anche per un valore di matrice. L'esempio seguente interroga la collection movies per selezionare tutti i documenti in cui l'array cast è esattamente uguale all'array specificato o l'array cast contiene un elemento uguale all'array **["Tom Cruise", "Jennifer Connelly", "Val Kilmer"** `]`.

```
db.movies.find( { cast: { $eq: [ "Tom Cruise", "Jennifer Connelly", "Val Kilmer" ]} } )
```

https://www.html.it/pag/52986/estrazione-dei-dati-find/

Operatori logici (AND/ OR)

Una query composta può specificare condizioni per più di un campo nei documenti della raccolta. Gli operatori logici forniti da MongoDB per la ricerca sono quelli della seguente tabella

Nome	Descrizione
$and	Unisce le clausole di query con una logica AND. Restituisce tutti i documenti che soddisfano le condizioni di entrambe le clausole.
$not	Inverte l'effetto di un'espressione di query e restituisce documenti che *non* corrispondono all'espressione di query.
$nor	Unisce le clausole di query con una logica NOR. Restituisce tutti i documenti che non corrispondono a entrambe le clausole.
$or	Unisce le clausole di query con una logica OR. Restituisce tutti i documenti che soddisfano le condizioni di una delle due clausole.

Facciamo un esempio. Per restituire i film della collection movies che sono usciti nel 2022 **e** durano più di 120 minuti **o** hanno come genere "Avventura" si deve dare il deguente comando:

```
use archivio_film
db.movies.find( {
    anno: 2022,
    $or: [ { durata: { $gte: 120 } }, { genere: "Avventura" } ]
} )
```

Il **$not**, essendo un operatore unario per definizione, accetta un solo parametro e si inserisce sempre dopo il nome del campo:

```
db.movies.find(  { durata: { $not: { $gte: 120 } } } )
```

Operatori sullo schema dei dati

Tutti i filtri che abbiamo visto finora si basano sul valore dei campi. Se un documento non ha il campo indicato nel filtro, esso non verrà preso in considerazione e quindi non verrà restituito come risultato di find.

Si possono filtrare i documenti richiedendo che esista espressamente un certo campo, oppure che esso non esista. Gli operatori sullo schema dei dati forniti da MongoDB per la ricerca sono quelli della seguente tabella

Nome	Descrizione
$exists	Corrisponde ai documenti che hanno il campo specificato.
$type	Seleziona i documenti se un campo è del tipo specificato.

La query seguente restituisce solo i film per i quali il campo durata non è specificato:

```
db.books.find(  { durata: { $exists: false } } )
```

Criteri di proiezione

Per impostazione predefinita, le query in MongoDB restituiscono tutti i campi nei documenti corrispondenti. Per limitare la quantità di dati che MongoDB invia alle applicazioni, si può includere un documento di proiezione per specificare o limitare i campi da restituire.

Se non si specifica un documento di proiezione, il metodo **db.collection.find()** restituisce tutti i campi nei documenti corrispondenti.

L'esempio seguente restituisce tutti i campi di tutti i documenti nella collection inventory dove status è uguale a "A":

```
db.inventory.find( { status: "A" } )
```

L'operazione corrisponde alla seguente istruzione SQL:

```
SELECT * from inventory WHERE status = "A"
```

Restituire i campi specificati e solo il campo _id

Una proiezione può includere esplicitamente diversi campi impostando <field>a 1 nel documento di proiezione. L'operazione seguente restituisce tutti i documenti in cui il campo status vale "A". Nel set di risultati, vengono restituiti solo i campi selezionati e, per impostazione predefinita, il campo _id.

```
db.inventory.find( { status: "A" }, { item: 1, status: 1 } )
```

L'operazione corrisponde alla seguente istruzione SQL:

```
SELECT _id, item, status from inventory WHERE status = "A"
```

Sopprimere il campo _id

È possibile rimuovere il campo _id dai risultati impostandolo su 0 nella proiezione, come nell'esempio seguente:

```
db.inventory.find( { status: "A" }, { item: 1, status: 1, _id: 0 } )
```

L'operazione corrisponde alla seguente istruzione SQL:

```
SELECT item, status from inventory WHERE status = "A"
```

Escludere dei campi

Invece di elencare i campi da restituire nel documento corrispondente, è possibile utilizzare una proiezione per escludere campi specifici. Il seguente esempio che restituisce tutti i campi tranne i campi **status** e instock nei documenti corrispondenti:

```
db.inventory.find( { status: "A" }, { status: 0, instock: 0 } )
```

Proiezione su documenti specifici incorporati in un array

Per visualizzare campi specifici all'interno di documenti incorporati in una matrice si deve utilizzare la notazione con il punto.

L'esempio seguente specifica una proiezione da restituire:

- Il campo **_id** (restituito per impostazione predefinita),
- Il campo **item**,
- Il campo **status**,

- Il campo **qty** nei documenti incorporati nell'array **instock**.

```
db.inventory.find(
  { status: "A" },
  { item: 1, status: 1, "size.uom": 1 }
)
```

è possibile anche specificare i campi incorporati utilizzando il modulo nidificato

```
db.inventory.find(
  { status: "A" },
  { item: 1, status: 1, size: { uom: 1 } }
)
```

Eliminare campi specifici nei documenti incorporati

Per eliminare campi specifici in un documento incorporato si deve utilizzare la notazione con il punto per fare riferimento al campo incorporato nel documento di proiezione e impostarlo su 0.

L'esempio seguente specifica una proiezione per escludere il campo uom campo all'interno del documento incorporato **size**. Tutti gli altri campi vengono restituiti nei documenti corrispondenti:

```
db.inventory.find(
  { status: "A" },
  { "size.uom": 0 }
)
```

è possibile anche specificare i campi incorporati utilizzando il modulo nidificato

```
db.inventory.find(
  { status: "A" },
  { size: { uom: 0 } }
)
```

Visualizzazione documenti incorporati in un array

Usare la notazione punto per proiettare campi specifici all'interno di documenti incorporati in una matrice.

L'esempio seguente specifica una proiezione da restituire:

Il campo **_id** (restituito per impostazione predefinita),

- Il campo **item**,
- Il campo **status**,
- Il campo **qty** nei documenti incorporati nell'array **instock**.

```
db.inventory.find( { status: "A" }, { item: 1, status: 1, "instock.qty": 1 } )
```

Visualizzazione elementi specifici di un array

Per i campi che contengono array, MongoDB fornisce i seguenti operatori di proiezione per manipolare gli array:

- **$elemMatch**,
- **$slice**,
- **$**.

Operazioni di Update (aggiornamento)

La shell MongoDB fornisce i seguenti metodi per aggiornare i documenti in una raccolta:

- **db.collection.updateOne()** per aggiornare un singolo documento.
- **db.collection.updateMany()** per aggiornare più documenti.
- **db.collection.replaceOne()** per sostituire un documento.

Per aggiornare un documento, MongoDB fornisce l'operatore di aggiornamento **$set** per modificare i valori dei campi.

db.collection.updateOne()

Esegue l'aggiornamento di un singolo documento all'interno di una collection

Per modificare il documento relativo al film "Top Gun Maverick" all'ìinterno della collection movies del database archivio_film e portare la durata a 125 minuti si deve dare il seguente comando

```
use archivio_film

db.movies.updateOne(
   { titolo: "Top Gun Maverick" },
   {  $set: { durata: 125 } }
)
```

db.collection.updateMany()

Esegue l'aggiornamento di tutti i documenti che corrispondono a un filtro specificato.

Il codice di questo esempio aggiorna tutti i film che hanno una durata maggiore o uguale a 135 minuti impostando il campo proiezioni a 3 e il campo extratime a vero. Se i campi non esistono li crea nei documenti aggiornati.

```
use archivio_film

db.movies.updateMany (
{ durata: { $gt: 135 } },
  {
    $set: { proiezioni : 3, extratime: true }
  }
)
```

db.collection.replaceOne()

Sostituisce l'intero contenuto di un documento ad eccezione del campo _id, si deve passare un documento completamente nuovo come secondo argomento.

Esempio

Il codice di questo esempio sostituisce il primo documento relativo al film "Elvis" con il documento relativo al film "Jurassic World il Dominio" lasciando però inalterato il campo _id

```
use archivio_film
```

```
db.movies.replaceOne (
        { titolo: “Elvis”},
        {
           titolo: "Jurassic World il Dominio",
           genere: [ "Azione", "Avventura", “Fantascienza” ],
           durata: 146,
           anno: 2022,
           classificazione: “PG-13”,
           regista: "Colin Trevorrow",
           cast: [ "Chris Pratt", "Bryce Dallas Howard", "Sam Neill" ]
        }
)
```

Operazioni di Delete (cancellazione)

La shell MongoDB fornisce i seguenti metodi per eliminare i documenti da una raccolta:

- **db.collection.deleteMany()** per eliminare più documenti,
- **db.collection.deleteOne()** per eliminare un singolo documento,.

db.collection.deleteMany()

Elimina tutti i documenti da una collection che soddisfino uno o più criteri. Se non vengono passati criteri di selezione, vengono cancellati tutti i documenti della collection.

Esempio

Il codice di questo esempio elimina tutti i documenti della collection movies

```
use archivio_film
db.movies.deleteMany( {} )
```

Per eliminare tutti i documenti che soddisfino ad un criterio di eliminazione, si deve passare un parametro di filtro al metodo deleteMany().

Esempio

Il codice di questo esempio elimina tutti i documenti della collection movies in cui l’array cast contiene “Tom Hanks”

```
use archivio_film
db.movies.deleteMany( { cast: "Tom Hanks" } )
```

db.collection.deleteOne()

Esempio

Il codice di questo esempio elimina il primo documento della collection movies in cui l’array cast contiene “Tom Cruise”

```
use archivio_film
db.movies.deleteOne( { cast: "Tom Cruise" } )
```