20/01/24

SPECIFICHE MONGODB

Campari Mirko

master.D

QUINDI COS'È MONGO DB?

MONGODB È UN DATABASE NOSQL ORIENTATO AI DOCUMENTI, CHE OFFRE DIVERSE FUNZIONI PER LA GESTIONE E IL RECUPERO DEI DATI.

ECCO UNA LISTA DELLE QUATTRO FUNZIONI PIÙ COMUNI DI MONGODB ABBIAMO GIÀ AFFRONTATO LA BASE TEORICA UTILE ALLA LORO COMPRENSIONE:

CREATE, READ, UPDATE, DELETE

master.D

CRUD Operations (Create, Read, Update, Delete):
Queste sono le operazioni fondamentali in MongoDB.

- **Create: Aggiunge nuovi documenti al database. In MongoDB, puoi utilizzare comandi come insertOne() o insertMany() per inserire documenti in una collezione.**

- **Read: Recupera i dati dal database. Utilizzando find() o findOne(), puoi cercare documenti che soddisfano determinati criteri.**

- **Update: Modifica i dati esistenti nel database. Con updateOne(), updateMany(), o replaceOne(), puoi cambiare i valori dei campi dei documenti esistenti.**

- **Delete: Rimuove i documenti dal database. I comandi deleteOne() e deleteMany() permettono di eliminare documenti in base ai criteri specificati.**

master.D

INDEXING:

MONGODB UTILIZZA GLI INDICI PER MIGLIORARE LA VELOCITÀ DI RICERCA DEI DOCUMENTI ALL'INTERNO DELLE COLLEZIONI. GLI INDICI SUPPORTANO L'ESECUZIONE EFFICIENTE DELLE QUERY;

SENZA INDICI, MONGODB DEVE ESEGUIRE UNA SCANSIONE DI TUTTI I DOCUMENTI DI UNA COLLEZIONE PER TROVARE QUELLI CHE CORRISPONDONO ALLA QUERY.

master.D

AGGREGATION:

QUESTA FUNZIONE PERMETTE DI ELABORARE I DATI E RESTITUIRE I RISULTATI CALCOLATI.

LE PIPELINE DI AGGREGAZIONE RAGGRUPPANO I DOCUMENTI E POSSONO ESEGUIRE UNA VARIETÀ DI OPERAZIONI SUI DATI AGGREGATI PER RESTITUIRE UN RISULTATO CALCOLATO.

È SIMILE ALLE FUNZIONI DI GRUPPO IN SQL.

master.D

REPLICATION:

MONGODB PUÒ FORNIRE ALTA DISPONIBILITÀ CON SET DI REPLICHE.

UN SET DI REPLICHE È UN GRUPPO DI ISTANZE MONGODB CHE MANTENGONO LO STESSO SET DI DATI.

LA REPLICA FORNISCE RIDONDANZA E AUMENTA LA DISPONIBILITÀ DEI DATI CON FAILOVER AUTOMATICO E RECUPERO DI DATI.

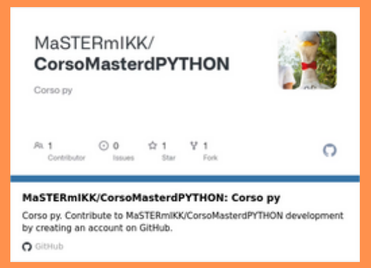
master.D

ANDREMO ORA A VEDERE UN ESEMPIO DI CODICE BASE IN PYTHON CHE UTILIZZA PYMONGO, UNA LIBRERIA POPOLARE PER LAVORARE CON MONGODB, PER DIMOSTRARE LE OPERAZIONI CRUD (CREATE, READ, UPDATE, DELETE),

L'INDICIZZAZIONE, L'AGGREGAZIONE E LA REPLICA.

ASSICURATI DI AVERE MONGODB INSTALLATO E IN ESECUZIONE SUL TUO SISTEMA E DI AVER INSTALLATO PYMONGO TRAMITE PIP INSTALL PYMONGO.

master.D




```python
from pymongo import MongoClient

# Connessione al server MongoDB (assicurati che MongoDB sia in esecuzione su localhost porta 27017)
client = MongoClient('localhost', 27017)

# Seleziona o crea un database chiamato 'mydatabase'
db = client['mydatabase']

# Seleziona o crea una collezione chiamata 'mycollection'
collection = db['mycollection']

# 1. Operazioni CRUD
# Create: Inserisce un nuovo documento nella collezione
doc = {"name": "Alice", "age": 30, "city": "New York"}
collection.insert_one(doc)

# Read: Trova un documento nella collezione
query = {"name": "Alice"}
user_doc = collection.find_one(query)
print(user_doc)

# Update: Aggiorna un documento
new_values = {"$set": {"age": 31}}
collection.update_one(query, new_values)

# Delete: Cancella un documento
collection.delete_one(query)

# 2. Indicizzazione
# Crea un indice sulla base del campo 'name' per velocizzare le query su quel campo
collection.create_index("name")

# 3. Aggregazione

# Esempio di pipeline di aggregazione per raggruppare documenti
pipeline = [
    {"$group": {"_id": "$city", "total": {"$sum": 1}}}
]
results = collection.aggregate(pipeline)
for result in results:
    print(result)

# Nota: La replica non è qualcosa che possiamo dimostrare con un semplice script Python.
# Richiede una configurazione del server MongoDB e può essere impostata seguendo la documentazione ufficiale.

# Chiudi la connessione
client.close()
```

master.D

Spiegazione DELL’ESEMPIO PRECEDENTE:

- MongoClient: Si connette al server MongoDB. Nell'esempio, si connette a localhost sulla porta standard 27017.
- db = client['mydatabase']: Seleziona il database mydatabase. Se non esiste, verrà creato quando si inserisce il primo documento.
- collection = db['mycollection']: Seleziona la collezione mycollection. Come per il database, se non esiste, verrà creata al primo inserimento.
- CRUD Operations: Sono mostrate operazioni di inserimento, lettura, aggiornamento e cancellazione di documenti. I documenti in MongoDB sono rappresentati come dizionari in Python.
- create_index: Crea un indice per velocizzare le ricerche. Gli indici sono cruciali per le prestazioni su grandi set di dati.
- aggregate: Esegue una pipeline di aggregazione. Nell'esempio, raggruppa i documenti per città e conta quanti utenti ci sono per ciascuna città.

Teniamo in conto durante questo capitolo che librerie più usate per lavorare con MongoDB variano a seconda del linguaggio di programmazione.
Ecco una lista delle librerie comunemente utilizzate in alcuni dei linguaggi di programmazione più popolari:

Python: pymongo

- È la libreria Python ufficiale per MongoDB. Fornisce un'interfaccia completa per lavorare con MongoDB e include funzionalità come la gestione delle connessioni, operazioni CRUD, indicizzazione e aggregazione.

JavaScript (Node.js): mongoose e mongodb

- mongodb: È il driver ufficiale Node.js per MongoDB. Fornisce funzionalità di basso livello per interagire con il database.
- mongoose: È un ODM (Object Document Mapping) che si costruisce sul driver mongodb. Offre un'interfaccia di alto livello con funzionalità aggiuntive come la validazione dei dati, la creazione di schemi e la gestione delle relazioni tra i dati.

master.d

Java: MongoDB Java Driver

- **È il driver ufficiale di MongoDB per Java. Fornisce un'API per lavorare con MongoDB, inclusi metodi per operazioni CRUD, aggregazione, gestione delle transazioni e altro ancora.**

C#: MongoDB.Driver

- **Questo è il driver ufficiale per MongoDB in C#. Fornisce un'interfaccia .NET per interagire con MongoDB e include supporto per LINQ, operazioni asincrone e altro.**

PHP: mongodb

- **Il driver ufficiale per MongoDB in PHP. Fornisce un'interfaccia per operare con MongoDB e supporta le operazioni CRUD, la gestione delle sessioni, l'aggregazione, ecc.**

master.D

Ruby: mongo

- **Il driver ufficiale MongoDB per Ruby. Offre funzionalità per interagire con MongoDB, inclusa la gestione delle connessioni, le operazioni CRUD e la pipeline di aggregazione.**

Go: mongo-go-driver

- **Questo è il driver ufficiale per MongoDB in Go. Fornisce un'API idiomatica per Go per interagire con MongoDB e supporta tutte le operazioni standard di MongoDB.**

Queste librerie sono comunemente utilizzate nelle loro rispettive comunità di sviluppatori e offrono un'ampia gamma di funzionalità per interagire con MongoDB.

La scelta della libreria dipende dal linguaggio di programmazione e dalle esigenze specifiche del progetto.

master.D

Indicizzazione

L'indicizzazione in MongoDB serve a ottimizzare le prestazioni delle query. Un indice consente a MongoDB di trovare rapidamente i documenti senza dover esaminare ogni documento della collezione.

```
from pymongo import MongoClient

# Connessione al server MongoDB
client = MongoClient('localhost', 27017)

# Seleziona il database e la collezione
db = client['mydatabase']
collection = db['mycollection']

# Creazione di un indice
collection.create_index([('name', 1)])  # Creazione di un indice ascendente sul campo 'name'

# Verifica degli indici esistenti
print(list(collection.index_information()))
```

master.D

Indicizzazione

In questo esempio, creiamo un indice sul campo name della collezione mycollection.

L'indice è ascendente, come indicato dal valore 1.

Gli indici possono essere anche discendenti (-1). index_information() fornisce informazioni sugli indici esistenti nella collezione.

master.d

Aggregazione

La pipeline di aggregazione in MongoDB consente di trasformare e combinare i dati in molteplici modi. Ecco un esempio di una semplice pipeline di

```
# Pipeline di aggregazione
pipeline = [
    {"$match": {"city": "New York"}},
# Fase di filtraggio: seleziona documenti con city = "New York"

    {"$group": {"_id": "$city", "averageAge": {"$avg": "$age"}}}
# Fase di raggruppamento: calcola l'età media
]

# Esecuzione della pipeline
aggregation_result = collection.aggregate(pipeline)

# Stampa dei risultati
for doc in aggregation_result:
    print(doc)
```

master.D

MaSTERmIKK/
CorsoMasterdPYTHON

Corso py

1 Contributor 0 Issues 1 Star 1 Fork

MaSTERmIKK/CorsoMasterdPYTHON: Corso py

Corso py. Contribute to MaSTERmIKK/CorsoMasterdPYTHON development by creating an account on GitHub.

GitHub

master.D

SINTESI PARZIALE SU MONGODB:

ATTRAVERSO LA LIBRERIA PYMONGO, GLI SVILUPPATORI PYTHON POSSONO FACILMENTE INTERAGIRE CON MONGODB, SFRUTTANDO OPERAZIONI CRUD, AGGREGAZIONI E ALTRE POTENTI FUNZIONALITÀ.
TUTTAVIA, COME CON QUALSIASI TECNOLOGIA, È ESSENZIALE COMPRENDERE E NAVIGARE ATTRAVERSO LE SUE SFIDE E LIMITAZIONI, COME QUESTIONI DI SCALABILITÀ, SICUREZZA E CONSISTENZA DEI DATI.

NEL COMPLESSO, MONGODB È UNA SOLUZIONE ROBUSTA E SCALABILE CHE, SE UTILIZZATA CORRETTAMENTE E IN CONTESTI APPROPRIATI, PUÒ FORNIRE PRESTAZIONI ECCEZIONALI E SEMPLIFICARE NOTEVOLMENTE LO SVILUPPO DI APPLICAZIONI.

SEGUE PARTE 2 ---> 23/01