MongoDB - Esercitazione

# Query semplici (find)

## Restaurants

1. Visualizzare tutti i ristoranti.
2. Visualizzare quartiere (borough) e tipo di cucina (cuisine) di tutti i ristoranti.
3. **Visualizzare quartiere (borough) e tipo di cucina (cuisine) di tutti i ristoranti, ma senza \_id.**
4. Visualizzare quartiere (borough), tipo di cucina (cuisine) e via (address.street) di tutti i ristoranti.
5. Visualizzare iI ristorante il cui zipcode è 11225
6. **Visualizzare i ristoranti il cui tipo di cucina è Hamburgers**
7. **Visualizzare i ristoranti il cui tipo di cucina NON è Hamburgers**
8. Visualizzare i ristoranti il cui tipo di cucina è tra Hamburgers, Bakery o Irish
9. Visualizzare i ristoranti il cui tipo di cucina NON è tra Hamburgers, Bakery o Irish
10. Visualizzare i ristoranti il cui tipo di cucina non esiste

## YelpBusiness

1. **Visualizzare tutti i business che hanno ricevuto almeno 10 recensioni (review\_count) E si trovano in Arizona o in Nevada (state = NV o AZ)**
2. **Visualizzare tutti i business che hanno ricevuto almeno 10 recensioni (review\_count) O si trovano in Arizona o in Nevada (state = NV o AZ)**
3. Visualizzare tutti i business che hanno Grocery tra le categorie
4. Visualizzare tutti i business che hanno Grocery E Barbeque tra le categorie
5. Visualizzare tutti i business che hanno Grocery O Barbeque tra le categorie
6. Visualizzare tutti i business che hanno SOLAMENTE Grocery E Barbeque tra le categorie (0 risultati)
7. Visualizzare tutti i business che hanno 5 categorie
8. Visualizzare tutti i business che hanno 5 categorie e la quinta categoria è Food
9. Visualizzare le prime due categorie di ogni business
10. Visualizzare unicamente le categorie dei business del Nevada
11. Visualizzare i valori distinti del campo city
12. Visualizzare i valori distinti del campo city nello stato del Nevada
13. Visualizzare i valori distinti dell’array categories

## Games

1. **Visualizzare le partite disputate nel 2010. (1239 risultati)**
   * Formato data: "anno-mese-giorno"
2. Visualizzare le partite in cui una delle squadre ha totalizzato almeno 160 punti. (6 risultati)
3. **Visualizzare le partite in cui la squadra di casa ha perso. (12197 risultati)**
4. Visualizzare le partite in cui ha giocato Michael Jordan. (990 risultati)
5. **Visualizzare le partite in cui Michael Jordan ha segnato più di 60 punti. (4 risultati)**
6. Visualizzare le prime 10 partite memorizzate nella collection. (10 risultati)
7. Visualizzare le prime 10 partite in ordine di data. (10 risultati)
8. Visualizzare le prime 10 partite del 2010 in ordine di data. (10 risultati)
9. Visualizzare le seconde 10 partite (dalla 11° del 2010 in ordine di data). (10 risultati)

# Indici di testo

## Restaurants

1. Creare un indice di testo sul campo nome
2. Cercare i ristoranti che hanno il termine “baker” nel nome; verificare i risultati

db.restaurants.find({$text: {$search: "baker" }})

1. Cercare i ristoranti che hanno il termine “bake” nel nome; verificare i risultati

## YelpBusiness

1. Creare un indice di testo sul campo categories
2. Contare i business che utilizzano il termine “service”
3. Contare i business che utilizzano il termine “services”
4. Contare i business che utilizzano i termini “professional” o “service”
5. Contare i business che utilizzano i termini “professional” e “service”
6. Contare i business che utilizzano il termine “professional” senza “service”

# Query complesse (aggregate: $project e $match)

## Yelpbusiness

1. **Restituire, per ogni business, un campo totalStars ottenuto moltiplicando stars per review\_count**
2. Restituire, per ogni business, un campo totalStars2 ottenuto moltiplicando stars per review\_count e sottraendo il campo is\_open al risultato
3. **Restituire, per ogni business, un campo isNevada che indichi se il business si trova o meno nello stato del Nevada (operatore $eq)**
4. Restituire, per i soli business che si trovano in Nevada, il campo totalStars come definito in query 1
5. Restituire, per ogni business, un campo totalStars3 ottenuto moltiplicando stars per review\_count se il business si trova nel Nevada, oppure dividendo stars per review\_count in caso contrario (operatore $cond)

## Games

1. Restituire (in campi separati) l’anno, il mese ed il giorno di ogni partita
2. **Restituire un campo che indichi quanti anni fa è stata disputata ciascuna partita**

**(sottrarre l’anno di *new Date()* dall’anno della partita)**

1. Restituire un campo che indichi quanti giorni fa è stata disputata ciascuna partita

(sottrarre *new Date()* dalla data della partita e trasformare da millisecondi a giorni)

1. Restituire, per ogni partita, un campo “team1” ed un campo “team2” che contengono rispettivamente il contenuto del primo e del secondo oggetto all’interno dell’array *teams*

db.games.aggregate([{  
 $project: {  
 "team1": {$arrayElemAt: ["$teams", 0]},  
 "team2": {$arrayElemAt: ["$teams", 1]}­­  
 }  
},{  
 $project: {  
 "diffPunti": {   
 $abs: {   
 $subtract: ["$team1.score","$team2.score"]   
 }  
 },  
 "punti1": "$team1.score",  
 "punti2": "$team2.score"  
 }   
}])

1. Data la query precedente, restituire, per ogni partita, la differenza tra i punteggi delle due squadre (utilizzare l’operatore “$abs”)  
   db.games.aggregate([{  
    $project: {  
    "team1": {$arrayElemAt: ["$teams", 0]},  
    "team2": {$arrayElemAt: ["$teams", 1]}­­  
    }  
   },{  
    $project: {  
    "diffPunti": {   
    $abs: {   
    $subtract: ["$team1.score","$team2.score"]   
    }  
    },  
    "punti1": "$team1.score",  
    "punti2": "$team2.score"  
    }   
   }])
2. Data la query precedente, restituire anche un campo booleano che indichi se la partita si è conclusa con un punto di differenza (true) o con più punti di differenza (false)
3. Data la query precedente, filtrare solo le partite in cui la differenza è di un unico punto
4. **Data la query precedente, filtrare solo le partite in cui la differenza è di un unico punto E in cui ha vinto la squadra di casa**

## Restaurants

1. Restituire un unico campo “fullAddress” che metta insieme (nell’ordine indicato) i campi zipcode, street e building contenuti dentro ad address
2. Restituire i primi 3 caratteri del campo borough
3. Restituire i primi 3 caratteri del campo borough in lettere maiuscole

# Query complesse (aggregate: $group e $sort)

## Restaurants

1. Raggruppare per borough
2. Raggruppare per borough e contare i ristoranti
3. Raggruppare per cuisine e contare i ristoranti; ordinare il risultato alfabeticamente
4. **Raggruppare per cuisine e contare i ristoranti; ordinare il risultato per numero decrescente di ristoranti**

## Yelp-business

1. Raggruppare per state e calcolare il punteggio medio (stars)
2. Raggruppare per city e per state e calcolare il punteggio medio (stars)
3. **Raggruppare per state e calcolare il punteggio medio (stars) e la somma di review ricevute**
4. Raggruppare per city del Nevada e calcolare il punteggio medio (stars) e la somma di review ricevute
5. Data la query 24, mantenere solo le city che hanno almeno 4.5 di punteggio medio
6. **Data la query 24, mantenere solo le city che hanno almeno 4.5 di punteggio medio E almeno 100 review**
7. Raggruppare per state ed individuare per ciascuno il punteggio più elevato
8. Raggruppare per state ed individuare per ciascuno il punteggio più elevato, ma senza utilizzare l’operatore $max
9. Raggruppare per state ed individuare per ciascuno l’elenco distinto di città (usando l’operatore $addToSet)

# Query complesse (aggregate: $unwind)

## Games

1. **Per ogni partita, creare un campo idPartita che indichi la data di svolgimento e le squadre che hanno giocato (ad esempio, “1985-10-25: Chicago Bulls vs Cleveland Cavaliers)**
   * non richiede l’uso di $unwind
   * $dateToString: {format: "%Y-%m-%d", date: "$date"}
2. Eseguire l’unwind dell’array teams
3. Eseguire l’unwind dell’array teams e raggruppare per \_ID della partita per ottenere il totale di punti segnati
4. **Come la query precedente, ma raggruppare per idPartita (come definito in query 30)**
5. Eseguire l’unwind dell’array teams e raggruppare per anno per ottenere il totale di punti segnati
6. Dalla query precedente, raggruppare per anno e squadra per ottenere il totale di punti segnati
7. **Dalla query precedente, ordinare per anno e punti segnati (decrescente) per ottenere le classifiche annuali**
8. Eseguire l’unwind dell’array box e dell’array players
9. Dalla query precedente, raggruppare per player per ottenere il totale di punti segnati; ordinare il risultato per visualizzare per primi i migliori giocatori del’NBA

## Restaurants

1. Eseguire l’unwind dell’array grades
2. Dalla query 39, raggruppare per borough e ottenere lo score medio
3. Dalla query 39, raggruppare per cuisine e ottenere lo score medio
4. **Ripetere la query 41 considerando solo i voti assegnati nel 2015**
5. Dalla query 41, raggruppare per anno e contare il numero di voti assegnati

# Query complesse (aggregate: $lookup)

## Games & NBA2016

1. **Unire la collection games con la collection nba2016players**

db.games.aggregate([{  
 $unwind: "$box"  
},{  
 $unwind: "$box.players"  
},{  
 $group: {  
 "\_id": "$box.players.player"  
 }  
},{  
 $lookup: {  
 from: 'nba2016players',  
 localField: '\_id',  
 foreignField: 'name',  
 as: 'stat'  
 }  
}])

1. Dalla query precedente, escludere i giocatori per cui non sono state trovate statistiche
   * Attenzione: se non vengono trovate statistische, il nuovo campo sarà un array vuoto
2. Estendere la query precedente per visualizzare il nome del giocatore, il totale di punti fatti ed il valore suo contratto (contract.amount)

* Attenzione: questo richiede di “srotolare” l’array

1. **Estendere la query precedente per calcolare quanto “costa” un punto segnato da ciascun giocatore; ordinare i risultati sulla base di questo risultato**

* Attenzione: dividere il totale di punti fatti per il valore del contratto, moltiplicato per 1000

1. Unire la collection games con la collection nba2016teams

* Attenzione: in games, i nomi delle squadre concatenano un’indicazione geografica (“Los Angeles”) ed il nome vero e proprio (“Lakers”), che in nba2016teams sono separati nei campi “region” e “name”. Per fare il join, bisogna:
  + 1. Splittare il nome in games utilizzando *$split: [nomeCampo, separatore]*
    2. Considerare l’ultimo valore nell’array risultante usando *$slice: [array, valore]*
    3. Usare l’operatore $unwind per trasformare l’array risultante in un oggetto

1. **Estendere la query precedente per visualizzare il nome della squadra, il totale di punti fatti ed il costo medio di un biglietto (budget.ticketPrice)**

## YelpBusiness & Cities

1. Unire la collection yelpBusiness con la collection cities
2. **Calcolare, per ogni categoria, la popolazione media e ordinare in base a quest’ultima**