Data Base and Data Mining Group of Politecnico di Torino

NoSQL in MongoDB Compass

The practice purpose is to become familiar with **MongoDB Compass** tool. In this practice you are required to explore data and write some queries to retrieve data from a NoSQL database based on MongoDB.

1. Problem specifications

The database contains Car Sharing information divided into two main collections: Bookings and Parkings. The most relevant information for each collection is shown in Table 1 (Parkings) and 2 (Bookings).

Name	Type	Description
_id	objectid	Document identifier
address	string	Parking address of the vehicle
city	string	City location of the vehicle
engineType	string	Identifier of the engine type of the vehicle
exterior	string	String describing the external condition of the vehicle during the parking
final_date	date	Date and hour of the end of the parking period
fuel	int32	Fuel level (0-100) during the parking period
init_date	date	Date and hour of the beginning of the parking period
interior	string	String describing the internal condition of the vehicle during the parking
loc	coordinates	Coordinates of the parking location
plate	int32	Identifier of the vehicle's plate
smartphoneRequired	Boolean	Boolean value denoting if the smartphone is required to

		start/finish the parking
vendor	string	Company owner of the vehicle
vin	string	Identifier of the chassis of the vehicle

Table 1: **Parkings** database info.

Name	**Type**	**Description**
_id	objectid	Document identifier
car_name	string	Vehicle’s model
city	string	City location where the vehicle has been booked
distance	int32	Distance covered during the vehicle renting
driving	object	**distance** | int32 | Distance covered during the vehicle renting (in meters) **duration** | int32 | Duration of the renting (in seconds)
engineType	string	Identifier of the engine type of the vehicle
exterior	string	String describing the external condition of the vehicle during the renting
final_address	string	Address of the final position of the renting period
final_date	date	Date and hour of the end of the renting period
final_fuel	int32	Fuel level (0-100) at the end of the renting period
init_address	int32	Address of the starting position of the renting period
init_date	date	Date and hour of the beginning of the renting period
init_fuel	int32	Fuel level (0-100) at the beginning of the renting period
interior	string	String describing the internal condition of the vehicle during the renting
plate	int32	Identifier of the vehicle’s plate

smartphoneRequired	Boolean	Boolean value denoting if the smartphone is required to start/finish the parking		
vendor	string	Company owner of the vehicle		
walking	object	**distance**	int32	Walk distance to reach the vehicle (in meters).
		duration	int32	Duration of the walking trip to reach the vehicle (in seconds).

Table 2: **Bookings** database info.

2. Database Connection

Remote database connection

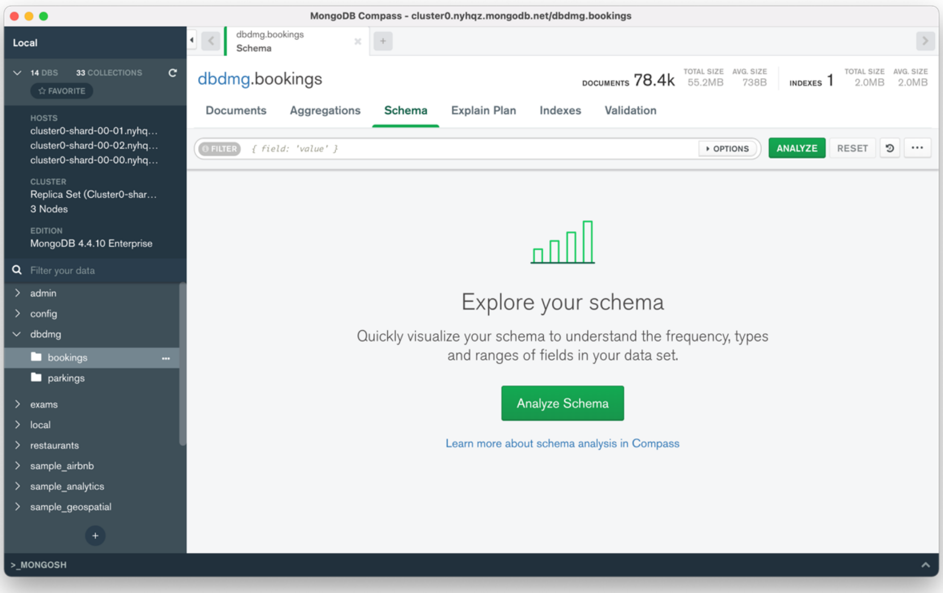
1. Download MongoDB Compass at https://www.mongodb.com/try/download/compass
2. Install and open the application.
3. Create a free cluster (if you do not have already one)
    a. create a MongoDB account (https://www.mongodb.com/cloud/atlas/register)
    b. Select *"Shared Cluster"* option (free cluster). The default settings are the ones set to get the account completely free.
    c. give a name to your cluster
    d. go to Database Access
        i. click on *"Add New Database User"*
        ii. select authentication method with password
        iii. fill in the form
        iv. leave all the default options
    e. configure remote access
        i. click on *"Network Access"*
        ii. click on *"Add IP Address"*
        iii. enter in the *Access List Entry* field 0.0.0.0/0
4. Get string connection from MongoDB Atlas server
    a. go on the "*Database Deployments"* page from the side menu
    b. click *"Connect"* near the newly created cluster
    c. select "*Compass*"
    d. copy the connection string
5. Paste the connection string into MongoDB Compass
6. Click on **Connect**

3. Create a database in MongoDB Compass

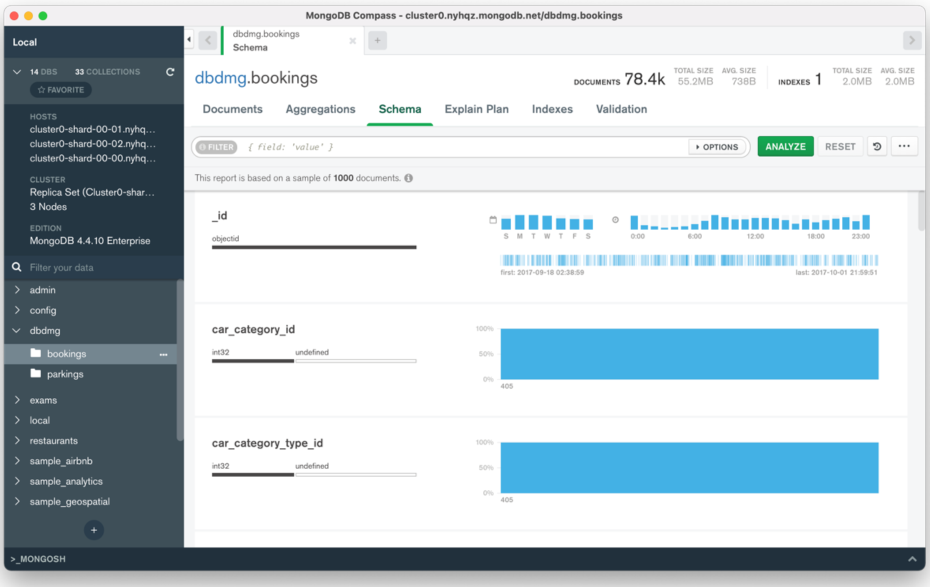
1. Click **Create Database**
    a. Assign a name to the database
    b. insert the name "Bookings" in the field "Collection Name"
    c. leave all the default values for the other options
2. Add a new collection
    a. click on *"Create Collection"* button

   b. enter the name "Parkings"
   c. leave all the default values for the other options
3. Download collection data from the website
4. Add data to each collection
   a. select one collection
   b. click *"Add Data"* button, then Import file
   c. select the file of the corresponding collection
   d. select Json type
   e. click *Import*
5. Select the Parkings collection
6. Go to the "Indexes" tab
7. Create a 2dsphere index on the loc field

4. Analyze the database using the Schema analyzer

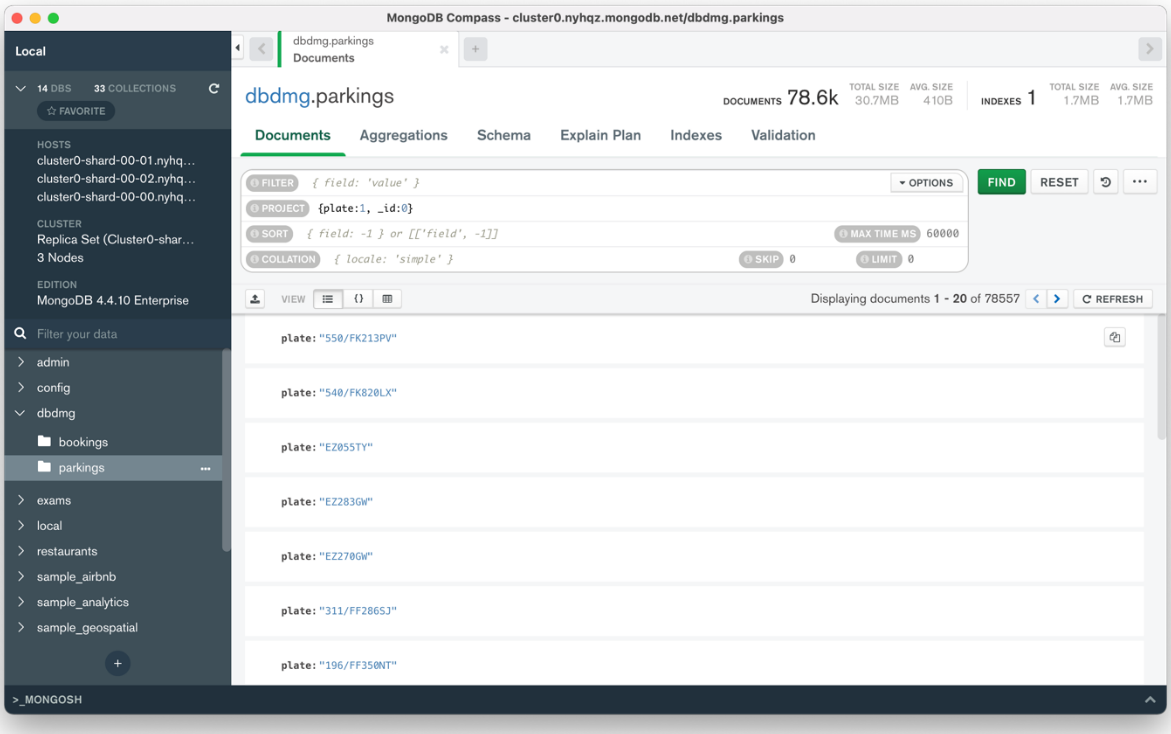


MongoDB Compass - cluster0.nyhqz.mongodb.net/dbdmg.bookings
Local
14 DBS 33 COLLECTIONS
FAVORITE
HOSTS
cluster0-shard-00-01.nyhq...
cluster0-shard-00-02.nyhq...
cluster0-shard-00-00.nyhq...
CLUSTER
Replica Set (Cluster0-shar...
3 Nodes
EDITION
MongoDB 4.4.10 Enterprise
Filter your data
admin
config
dbdmg
bookings
parkings
exams
local
restaurants
sample_airbnb
sample_analytics
sample_geospatial
dbdmg.bookings
Schema
DOCUMENTS 78.4k TOTAL SIZE 55.2MB AVG. SIZE 738B
INDEXES 1 TOTAL SIZE 2.0MB AVG. SIZE 2.0MB
Documents
Aggregations
Schema
Explain Plan
Indexes
Validation
FILTER
{ field: 'value' }
OPTIONS
ANALYZE
RESET
Explore your schema
Quickly visualize your schema to understand the frequency, types and ranges of fields in your data set.
Analyze Schema
Learn more about schema analysis in Compass
>_MONGOSH



1. (Bookings) Identify the most common percentage(s) of fuel level at the beginning of the renting period.
2. (Bookings) Identify the most common percentage(s) of fuel level at the end of the renting period.
3. (Parkings) Identify the time range(s) with most parking requests (start parking).
4. (Parkings) Identify the time range(s) with most booking requests (end parking).
5. (Parkings) Visualize on the map the vehicles having the fuel level lower than 5%.

5. Querying the database



```
MongoDB Compass - cluster0.nyhqz.mongodb.net/dbdmg.parkings

HOSTS
cluster0-shard-00-01.nyhq…
cluster0-shard-00-02.nyhq…
cluster0-shard-00-00.nyhq…
CLUSTER
Replica Set (Cluster0-shar…
3 Nodes
EDITION
MongoDB 4.4.10 Enterprise

dbdmg.parkings   DOCUMENTS 78.6k  TOTAL SIZE 30.7MB  AVG. SIZE 410B   INDEXES 1  TOTAL SIZE 1.7MB  AVG. SIZE 1.7MB

FILTER     { field: 'value' }
PROJECT    {plate:1, _id:0}
SORT       { field: -1 } or [['field', -1]]              MAX TIME MS 60000
COLLATION  { locale: 'simple' }            SKIP 0          LIMIT 0

Displaying documents 1 - 20 of 78557

plate: "550/FK213PV"
plate: "540/FK820LX"
plate: "EZ055TY"
plate: "EZ283GW"
plate: "EZ270GW"
plate: "311/FF286SJ"
plate: "196/FF350NT"
```

1. (Parkings) Find the plates and the parking addresses of the vehicles that begin the booking (end parking) after 2017-09-30 at 6AM.
   Hint: it is possible to use the function Date("<YYYY-mm-ddTHH:MM:ss>")
2. (Parkings) Find the addresses and the level of fuel of the vehicles that during the parking period had at least 70% of fuel level. Order the results according to descending value of fuel level.
3. (Parkings) Find the plate, the engine type and fuel level for 'car2go' vehicles (vendor) with good internal and external conditions.
4. (Bookings) For the renting that required a walking distance greater than 15 Km (to reach the vehicle), find the hour and the fuel level at the beginning of the renting period. Order results according to decreasing initial fuel level.

6. Data Aggregation

5. (Bookings) Group documents according to their fuel level at the end of the renting. For each group, select the average fuel level at the beginning of the renting period.
6. (Bookings) Select the average driving distance for each vendor. On average, for which vendor the users cover longer distances?

7. Bonus Queries

7. (Parkings) Find the vehicles parked less than a mile far from Piazza San Carlo (coordinates: 7.683016, 45.067764).
   Hint: use the operators $geoWithin and $centerSphere.
8. (Parkings) Repeat the query at the previous step using the coordinates of a place of personal interest in Turin (e.g. Politecnico di Torino) using Open Street Maps to find the exact coordinates (www.openstreetmap.org, inverse the coordinates order).