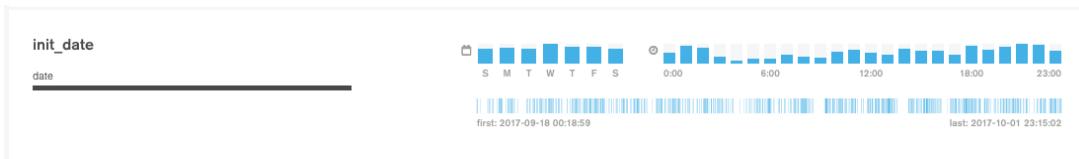


Business Intelligence per Big Data

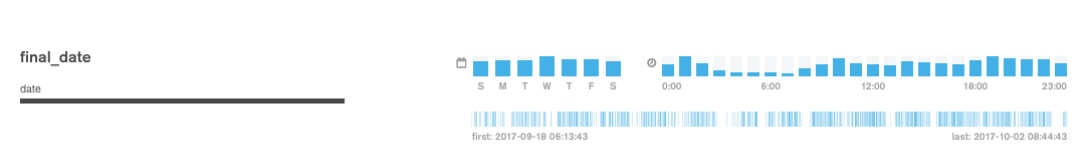
MongoDB Compass - Esercitazione n. 10 - Soluzioni

1. Analizzare la base dati con lo *schema analyzer* (Parkings)

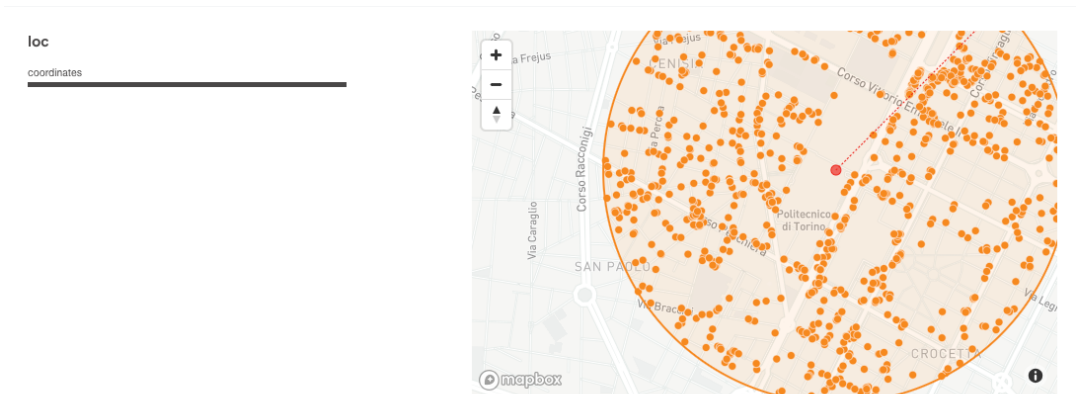
1. Identificare l'intervallo/gli intervalli orari con maggiore **richiesta di parcheggio (inizio stazionamento)** di veicoli.



2. Identificare l'intervallo/gli intervalli orari nei quali i veicoli **vengono noleggiati (fine stazionamento)** più di frequente.

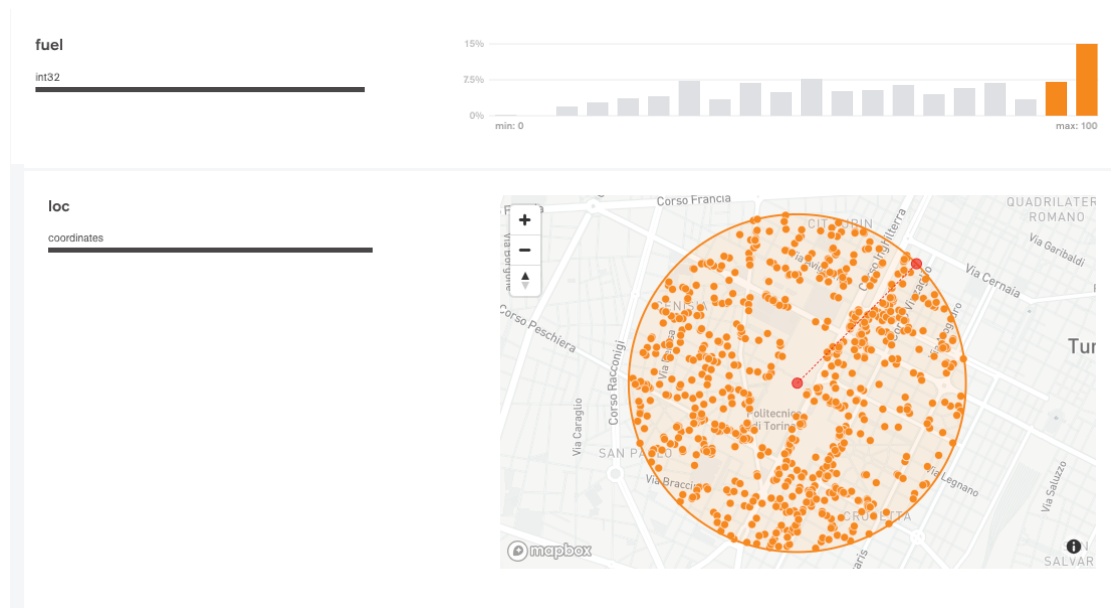


3. Filtrare sulla mappa una zona di interesse e analizzare l'intervallo/gli intervalli orari di **inizio noleggio (fine stazionamento)** più frequenti.

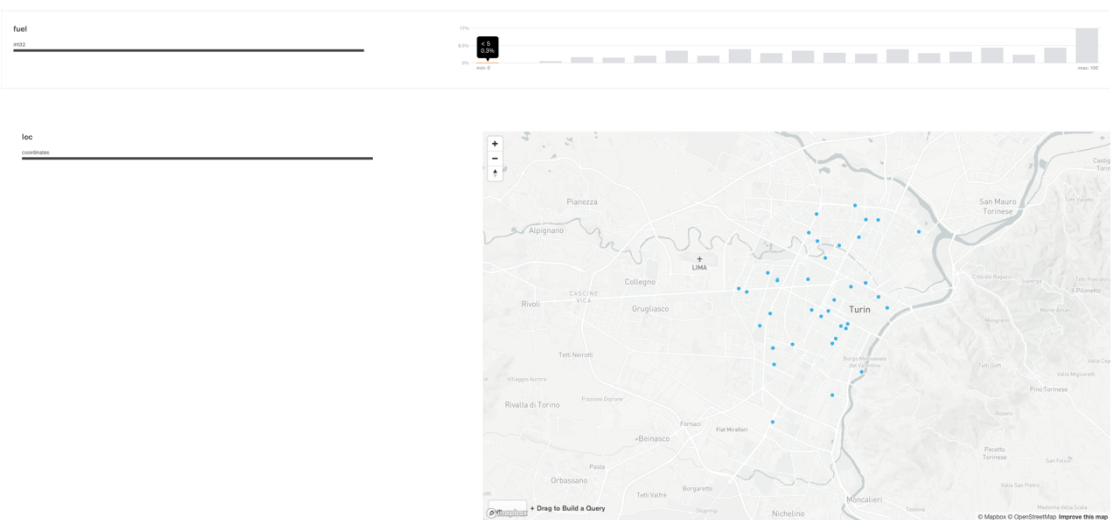


Analizzare l'intervallo/gli intervalli orari di inizio noleggio (fine stazionamento) più frequenti come fatto nell'esercizio 2.

- Per i veicoli filtrati al passo precedente, visualizzare solo quelli che hanno un livello di carburante residuo maggiore del 90%.

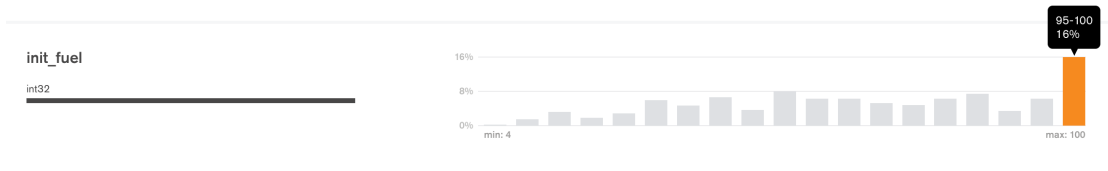


- Visualizzare sulla mappa i veicoli che hanno un livello di carburante residuo inferiore al 5%.

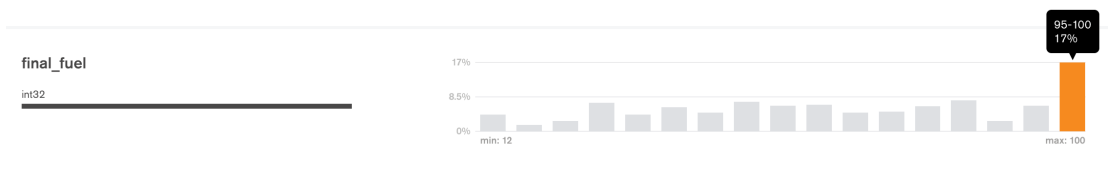


2. Analizzare la base dati con lo *schema analyzer* (Bookings)

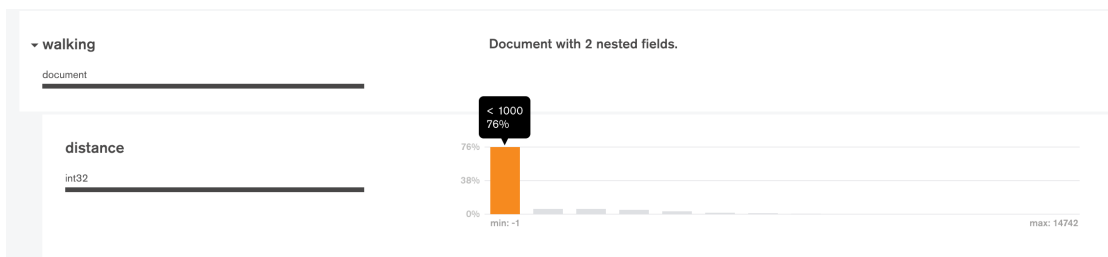
- (Bookings) Identificare la/le percentuali più frequenti di **livello di carburante a inizio noleggio (fine stazionamento)**.



2. (Bookings) Identificare la/le percentuali più frequenti di **livello di carburante a fine noleggio (inizio stazionamento)**.



3. (Bookings) Identificare il range di **distanza percorsa a piedi** più frequente per raggiungere il veicolo.



3. Interrogare la base dati (Parkings)

1. Trovare le targhe e gli indirizzi di parcheggio dei veicoli che hanno iniziato il noleggio (finito stazionamento) dopo le 06:00 del 30-09-2017.
(Hint: usare la funzione [ISODate\("YYYY-mm-ddTHH:MM:ss"\)](#))

`{final_date:{$gte: ISODate('2017-09-30T06:00:00')}}`

Project `{address:1, plate:1, _id:0}`
Max Time MS

Sort `{}`
Skip Limit

Collation `{ locale: 'simple' }`

Index Hint `{ field: -1 }`

1 - 20 of 10192

plate: "EZ266GW"
 address: "Via Luigi Pietracqua, 15, 10154 Torino TO"

plate: "EZ102TY"
 address: "Via Arbe, 2-12, 10136 Torino TO"

plate: "EZ233GW"
 address: "Via Arbe, 2-12, 10136 Torino TO"

2. Trovare gli indirizzi e il livello di carburante residuo per le auto che hanno avuto durante lo stazionamento almeno il 70% di carburante residuo e ordinare i risultati in base al loro livello di carburante decrescente.

`{fuel: {$gte: 70}}`

Project `{address:1, fuel:1, _id:0}`
Max Time MS

Sort `{fuel: -1}`
Skip Limit

Collation `{ locale: 'simple' }`

Index Hint `{ field: -1 }`

1 - 20 of 34371

fuel: 100
 address: "Corso Giulio Cesare, 2961, 10155 Torino TO"

fuel: 100
 address: "Corso Massimo d'Azeglio, 11, 10126 Torino TO"

fuel: 100
 address: "Via Giuseppe Maria Bonzanigo, 7-9, 10144 Torino TO"

3. Trovare la targa, tipo di motore e livello di carburante dei veicoli di 'car2go' che hanno buone condizioni interne ed esterne.

`{vendor: 'car2go', exterior: "GOOD", interior:"GOOD"}`

Project `{plate:1, engineType:1, fuel: 1, _id:0}`
Max Time MS

Sort `{ field: -1 } or [['field', -1]]`
Skip Limit

Collation `{ locale: 'simple' }`

Index Hint `{}`

1 - 20 of 48423

plate: "S40/FK820LX"
 fuel: 18
 engineType: "CE"

plate: "S35/FK815LX"
 fuel: 75
 engineType: "CE"

4. Interrogare la base dati (Bookings) – Documents e Aggregation Tab

1. (Bookings) Per i noleggi che hanno richiesto un percorso a piedi maggiore di 15 km per raggiungere il veicolo, visualizzare la data e l'orario di inizio noleggio e il livello di carburante a inizio noleggio. Visualizzare i risultati ordinati in base al livello di carburante iniziale decrescente.

```
{ "walking.distance": { "$gte": 15000 } }
```

Project: { init_date: 1, init_fuel: 1, _id: 0 }

Sort: { init_fuel: -1 }

Collation: { locale: 'simple' }

Index Hint: { field: -1 }

Max Time MS: 60000

Skip: 0 Limit: 0

EXPORT DATA

1 - 20 of 131

```
init_fuel : 100
init_date : 2017-09-20T04:05:38.000+00:00

init_fuel : 100
init_date : 2017-09-22T10:09:21.000+00:00

init_fuel : 100
init_date : 2017-09-27T07:27:55.000+00:00
```

2. (Bookings) Raggruppare i documenti in base al loro **livello di carburante a fine noleggio**. Per ogni gruppo visualizzare il **livello di carburante medio a inizio noleggio**.

```
$group = { _id: "$final_fuel",
           avg_init_fuel: { $avg: "$init_fuel" }
}
```

mp1.polito.it:8005 > bibd > Bookings

Documents 78.4K Aggregations Schema Indexes 1 Validation

Generate aggregation + ? Explain Export Run Options

Untitled - modified SAVE + CREATE NEW EXPORT TO LANGUAGE PREVIEW STAGES TEXT WIZARD

Stage 1 \$group

```
1
2 {
3   _id: "$final_fuel",
4   avg_init_fuel: {
5     $avg: "$init_fuel"
6   }
7 }
```

Output after \$group stage (Sample of 10 documents)

```
_id: 48
avg_init_fuel : 49.318734793187346

_id: 45
avg_init_fuel : 46.526315789473685
```

3. (Bookings) Visualizzare la **distanza media** percorsa nei noleggi per ciascun **fornitore del servizio**. In media con quale fornitore del servizio gli utenti percorrono una distanza maggiore?

```
$group = {_id: "$vendor",
          avg_distance: {$avg: "$distance" }
        }
```

The screenshot shows the MongoDB Atlas aggregation pipeline editor. The pipeline is named "\$group". The aggregation stage is defined as follows:

```
1 /**
2  * _id: The id of the group.
3  * fieldN: The first field name.
4  */
5 {
6   _id: "$vendor",
7   dist: {
8     $avg: "$distance"
9   }
10 }
```

The output after the \$group stage (Sample of 2 documents) is shown as follows:

_id: "car2go" dist: 1915.44424954083	_id: "enjoy" dist: 2430.8068234979614
---	--

4. (Bookings) Quali sono i **primi tre** veicoli che hanno percorso una maggiore distanza considerando tutti i noleggi? Visualizzarne la targa e il conteggio in ordine decrescente per totale distanza percorsa in tutti i noleggi.

Step 1: per ogni veicolo, calcolare la totale distanza percorsa in tutti i noleggi.

The screenshot shows the MongoDB Atlas aggregation pipeline editor. The pipeline is named "\$group". The aggregation stage is defined as follows:

```
1 /**
2  * _id: The id of the group.
3  * fieldN: The first field name.
4  */
5 {
6   _id: "$plate",
7   totDist: {
8     $sum: "$distance"
9   }
10 }
```

The output after the \$group stage (Sample of 10 documents) is shown as follows:

_id: "E2986DD" totDist: 333718	_id: "218/FF0653" totDist: 311061
-----------------------------------	--------------------------------------

Step 2: ordinare per totale distanza decrescente.

The screenshot shows the MongoDB Atlas aggregation pipeline editor. The pipeline is named "\$sort". The aggregation stage is defined as follows:

```
1 /**
2  * Provide any number of field/order pair
3  */
4 {
5   totDist: -1
6 }
```

The output after the \$sort stage (Sample of 10 documents) is shown as follows:

_id: "082/FF048NT" totDist: 428935	_id: "053/FF841KW" totDist: 388515
---------------------------------------	---------------------------------------

Step 3: visualizzare i primi 3 documenti.

The screenshot shows the MongoDB Atlas aggregation pipeline editor. The pipeline is named "\$limit". The aggregation stage is defined as follows:

```
1 /**
2  * Provide the number of
3  */
4 3
```

The output after the \$limit stage (Sample of 3 documents) is shown as follows:

_id: "082/FF048NT" totDist: 428935	_id: "053/FF841KW" totDist: 388515	_id: "053/FF841KW" totDist: 388515
---------------------------------------	---------------------------------------	---------------------------------------

5. (Bookings) Quanti sono i veicoli che, considerando tutti i noleggi, hanno percorso almeno 350000m?

Step 1: per ogni veicolo, calcolare la totale distanza percorsa in tutti i noleggi.

```

1 /**
2  * _id: The id of the group.
3  * fieldN: The first field name.
4  */
5 {
6   _id: "$plate",
7   totDist: {
8     $sum: "$distance"
9   }
10 }

```

Output after \$group stage (Sample of 10 documents)

_id: "E2986DD" totDist: 333718	_id: "218/FF0653" totDist: 311061
-----------------------------------	--------------------------------------

Step 2: verificare che la distanza totale sia almeno 350000m

```

1 /**
2  * query: The query in MQL.
3  */
4 {
5   totDist: { $gte: 350000 }
6 }

```

Output after \$match stage (Sample of 10 documents)

_id: "82/FF648NT" totDist: 428935	_id: "E2167TV" totDist: 353750
--------------------------------------	-----------------------------------

Step 3: contare il numero di veicoli

```

1 /**
2  * Provide the field name for the count.
3  */
4 'numero_veicoli'

```

Output after \$count stage (Sample of 1 document)

numero_veicoli: 21

6. (Bookings) Ripetere la query precedente considerando solamente i veicoli di proprietà 'car2go'.

Simile alla query precedente in cui si aggiunge lo Step 0 in cui andiamo a filtrare su *vendor*: "car2go".

```

1 /**
2  * query: The query in MQL.
3  */
4 {
5   vendor: "car2go"
6 }

```

Output after \$match stage (Sample of 10 documents)

_id: ObjectId('59befcd2ad8532ca60093a') init_fuel: 75 city: "Torino" walking: Object init_address: "Via Guido Reni, 26A, 10136 Torino TO" vendor: "car2go" final_time: 1505686390 driving: Object	_id: ObjectId('59bef1602ad8532ca6009fd') init_fuel: 37 city: "Torino" walking: Object init_address: "Via Andrea Sansovino, 35, Torino TO" vendor: "car2go" final_time: 1505688313 driving: Object
--	--

Interrogazioni Bonus (Parkings)

1. (Parkings) Quanti sono i parcheggi effettuati a meno di 1km da Piazza San Carlo (coordinate 7.683016, 45.067764).

Documents 78.6K Aggregations **Schema** Indexes 1 Validation

```
{loc: {$geoWithin: {$centerSphere: [[7.683016, 45.067764], 1/6378.1]]}}
```

Generate query + Reset Analyze Options

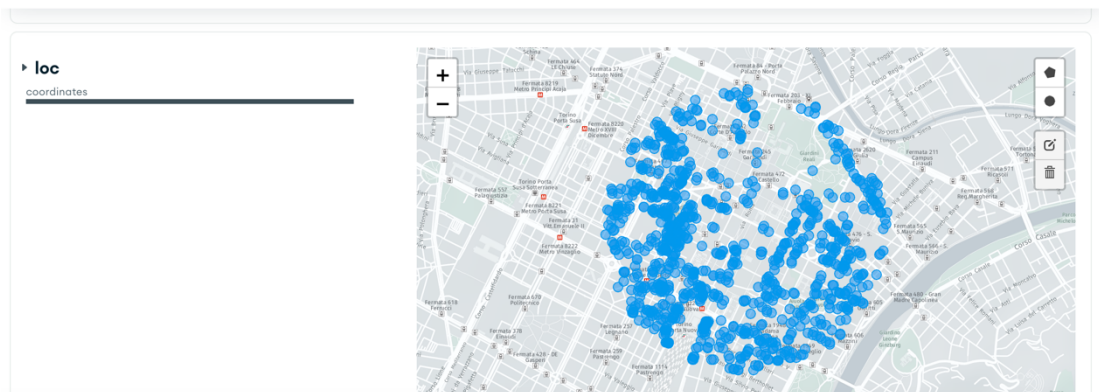
Project { field: 0 }

Sort { field: -1 } or [['field', -1]] Max Time MS 60000

Collation { locale: 'simple' } Skip 0 Limit 0

Index Hint {}

This report is based on a sample of 1000 documents. [Learn more](#)



2. Ripetere l'interrogazione al passo precedente con un punto di interesse personale nell'area metropolitana di Torino (e.g. indirizzo di casa) usando Open Street Maps per trovare le coordinate esatte (www.openstreetmap.org, invertire l'ordine delle coordinate).

Come la query precedente cambiando le coordinate con quelle di interesse.