**Domanda 1**

Per risolvere il quesito si può usare la query: db.getCollection('stations').aggregate( [ { $group: { \_id: '$extra.status', count: { $count: {} } } }, { $match: { $or: [ { \_id: 'offline' }, { \_id: 'online' } ] } } ]). Quindi otterremo che le stazioni che hanno extra.status “online” sono 33, mentre quelle che hanno extra.status ”offline” sono 28.

**Domanda 2**

Le stazioni che hanno extra.status diverso da “online” e “offline” sono 4. La query scelta è: db.getCollection('stations').find({$and:[{"extra.status":{$ne:"offline"}}, {"extra.status":{$ne:"online"}}]}).

**Domanda 3**

La query che permette di mantenere lo stesso risultato ma di mostrare solo il campo “extra.status” è: db.getCollection(‘stations').find({"extra.status": {$nin: ["online", "offline"]}}, {"extra.status": 1, \_id: 0}).Utilizzando il costrutto “nin” il risultato è il medesimo della questy utilizzando “and” e “ne”.

**Domanda 4**

La seguente query risponde alla domanda: db.getCollection('stations').find ({"extra.status": "online", "extra.score": {$gte:4}), {"name": 1, "\_id": 0}) .sort({"name":1}). Le stazioni che vengono richieste sono 18 che successivamente vengono mostrati in ordine alfabetico in output

**Domanda 5**

Soltanto la stazione di name=05 soddisfa i requisiti. la quale ha 0 “empty\_slots”, e 5 “free\_bikes”. La query per ottenere tale risultato è: db.getCollection('stations').find({"extra.status": "offline", $or:[ {"empty\_slot":{$gt:0}}, {"free\_bikes":{$gt:0}}]}), se si vuole visualizzare solo “name”, “empty\_slots” e “free\_bikes” si può usare: db.getCollection('stations').find({"extra.status": "offline", $or:[ {"empty\_slot":{$gt:0}}, {"free\_bikes":{$gt:0}}]}),{"name":1,"empty\_slots":1, "free\_bikes":1, "\_id": 0})

**Domanda 6**

Il numero totale di recensioni per tutte le stazioni è 15311. Si può trovare con: db.getCollection('stations').aggregate([ { $group: { \_id: null, totalReviews: { $sum: "$extra.reviews" } } } ]) .

**Domanda 7**

La query è: db.getCollection(‘stations').aggregate([{$group: { \_id: "$extra.score", count: { $count: {} } } }]).sort ({\_id:-1}). L’attributo -1 del sort serve per ordinare in ordine decrescente

**Domanda 8**

Una prima possible risposta, se si vogliono due medie separate, considerando separatemente i casi “online” e “offline” è : db.getCollection('stations').aggregate( [ { $group: { \_id: '$extra.status', average: { $avg: '$extra.score' } } }, { $match: { $or: [ { \_id: 'offline' }, { \_id: 'online' } ] } } ]). Le due medie saranno; 3.03 per i casi “offline”, e 3.84 per i casi “online”

Mentre se si vuole una sola media che considera sia i casi di “offline” che “online” la query è : db.getCollection('stations').aggregate( [ { $match: { $or: [ { 'extra.status': 'online' }, { 'extra.status': 'offline' } ] } }, { $group: { \_id: null, average: { $avg: '$extra.score' } } } ]). La media condivisa sarà quindi 3.47

**Domanda 9**

La query : db.getCollection('stations').aggregate( [ { $group: { \_id: { $cond: { if: { $eq: ['$free\_bikes', 0] }, then: 'Senza Bici', else: 'Con Bici' } }, gb: { $avg: '$extra.score' } } } ] ) ci permette di separare i documenti che hanno “free\_bikes”=0 e quelli con “free\_bikes”>0, ed effettuare la media sui loro “extra.score”. La media per le stazioni con almeno una bici è 3.87, mentre per quelle senza bici è 3.23

**Domanda 10**

Si può aggiungere alla risposta della domanda numero 9 un match prima del group. Query: db.getCollection('stations').aggregate( [ { $match: { 'extra.status': 'online' } }, { $group: { \_id: { $cond: { if: { $eq: ['$free\_bikes', 0] }, then: 'Senza Bici', else: 'Con Bici' } }, gb: { $avg: '$extra.score' } } } ]. La media per le stazioni con almeno una bici è 3.86, mentre per quelle senza bici è 3.74

**Domanda 11**

Si parte creando un indice geospaziale 2dsphere usando il comando: db.getCollection('stations').createIndex( {location: “2dsphere”} ) e poi la query : db.getCollection('stations').find({free\_bikes:{$gt:0}, location: { $near: { $geometry: { type: "Point", coordinates: [45.07456, 7.69463] } } } } ).limit(3), che restituirà le 3 stazioni con “free\_bikes”>0 più vicine alle coordinate date. I nomi delle 3 stazioni più vicine sono: “Palermo 2”, “Castello 1”, “San Frncesco da Paola”

**Domanda 12**

La query è: db.getCollection('stations').find({free\_bikes:{$gt:0}, location: { $near: { $geometry: { type: "Point", coordinates: db.getCollection('stations').findOne({name:”Politecnico 4”}).location.coordinates } } } } ).limit(3). La query annidata ci permette di recuperare le coordinate della prima stazione, mentre quella esterna trova le 3 stazioni più vicine a quella richiesta. I nomi delle 3 stazioni più vicine sono: “Politecnico 1”, “Politecnico 3”, “Tribunale”