Kevin De La Torre
Project 3

FIND Tutorial

2.

You can use `.pretty()` to make the output more readable.

List all the countries in the continent of "Eurasia".

```
db.world.find({
  continent: 'Eurasia'
}).pretty();
```

Run NoSQL | Format Answer | Restore Default | Toggle Font Size

Incorrect

Your answer:


```
{
    "_id" : ObjectId("5fb970b77cfb09a5969a12af"),
    "name" : "Armenia",
    "continent" : "Eurasia",
    "area" : 29743,
    "population" : 3017400,
    "gdp" : 9950000000,
    "capital" : "Yerevan",
    "tld" : ".am",
    "flag" : "//upload.wikimedia.org/wikipedia/commons/2/2f/Flag_of_Armenia.svg"
}
{
    "_id" : ObjectId("5fb970b77cfb09a5969a1334"),
    "name" : "Russia",
    "continent" : "Eurasia",
    "area" : 17125242,
    "population" : 146000000,
    "gdp" : 2029812000000,
    "capital" : "Moscow",
    "tld" : ".ru",
    "flag" : "//upload.wikimedia.org/wikipedia/commons/f/f3/Flag_of_Russia.svg"
}
```


Correct answer:


```
{
    "_id" : ObjectId("5fb68c8d0ed8f54602b46967"),
    "name" : "Armenia",
    "continent" : "Eurasia",
    "area" : 29743,
    "population" : 3017400,
    "gdp" : 9950000000,
    "capital" : "Yerevan",
    "tld" : ".am",
    "flag" : "//upload.wikimedia.org/wikipedia/commons/2/2f/Flag_of_Armenia.svg"
}
{
    "_id" : ObjectId("5fb68c8d0ed8f54602b469ec"),
    "name" : "Russia",
    "continent" : "Eurasia",
    "area" : 17125242,
    "population" : 146000000,
    "gdp" : 2029812000000,
    "capital" : "Moscow",
    "tld" : ".ru",
    "flag" : "//upload.wikimedia.org/wikipedia/commons/f/f3/Flag_of_Russia.svg"
}
```

3.

You can test numbers as well as strings.

Find the country with an area of exactly 43094.

```
1  db.world.find({area: 43094}).pretty();
```

Run NoSQL | Format Answer | Restore Default | Toggle Font Size

Incorrect

Your answer:


```
{
    "_id" : ObjectId("5fb970b77cfb09a5969a12d6"),
    "name" : "Denmark",
    "continent" : "Europe",
    "area" : 43094,
    "population" : 5634437,
    "gdp" : 314889000000,
    "capital" : "Copenhagen",
    "tld" : ".dk",
    "flag" : "//upload.wikimedia.org/wikipedia/commons/9/9c/Flag_of_Denmark.svg"
}
```


Correct answer:


```
{
    "_id" : ObjectId("5fb68c8d0ed8f54602b4698e"),
    "name" : "Denmark",
    "continent" : "Europe",
    "area" : 43094,
    "population" : 5634437,
    "gdp" : 314889000000,
    "capital" : "Copenhagen",
    "tld" : ".dk",
    "flag" : "//upload.wikimedia.org/wikipedia/commons/9/9c/Flag_of_Denmark.svg"
}
```

Using $gt

4.

You can use **$gt** (greater than) and **$lt** (less than) to compare numbers and strings.

Show each country with a population of over 250000000
Sort the results alphabetically.

You will need to use a projection to answer this question.

```
db.world.find({
  population: {
    $gt: 250000000
  }
}, {
  name: 1,
  _id: 0
}).sort({
  name: 1
}).pretty();
```

Run NoSQL | Format Answer | Restore Default | Toggle Font Size

Your answer:

```
{ "name" : "China" }
{ "name" : "India" }
{ "name" : "Indonesia" }
{ "name" : "United States" }
```

After S

5.

Greater than and less than comparisons can also be applied to strings.

List the countries that come after "S" in the alphabet.

```
db.world.find({
  name: {
    $gt: 'S'
  }
}, {
  name: 1,
  _id: 0
}).pretty();
```

Run NoSQL | Format Answer | Restore Default | Toggle Font Size

Your answer:

```
{ "name" : "Saint Kitts and Nevis" }
{ "name" : "Saint Lucia" }
{ "name" : "Saint Vincent and the Grenadines" }
{ "name" : "Samoa" }
{ "name" : "San Marino" }
{ "name" : "Sao TomÃ© and PrÃ•ncipe" }
{ "name" : "Saudi Arabia" }
{ "name" : "Senegal" }
{ "name" : "Serbia" }
{ "name" : "Seychelles" }
{ "name" : "Sierra Leone" }
{ "name" : "Singapore" }
{ "name" : "Slovakia" }
{ "name" : "Slovenia" }
{ "name" : "Solomon Islands" }
{ "name" : "Somalia" }
{ "name" : "South Africa" }
{ "name" : "South Korea" }
{ "name" : "South Sudan" }
{ "name" : "Spain" }
Type "it" for more
```

Name and Capital

6.

Find the name and capital cities for countries with a population of over 70 million.

```
db.world.find({
  population: {
    $gt: 70000000
  }
}, {
  name: 1,
  capital: 1,
  _id: 0
});
```

Run NoSQL | Format Answer | Restore Default | Toggle Font Size

Incorrect

Your answer:

```
{ "name" : "Bangladesh", "capital" : "Dhaka" }
{ "name" : "Brazil", "capital" : "BrasÃ•lia" }
{ "name" : "China", "capital" : "Beijing" }
{ "name" : "Egypt", "capital" : "Cairo" }
{ "name" : "Ethiopia", "capital" : "Addis Ababa" }
{ "name" : "Germany", "capital" : "Berlin" }
{ "name" : "India", "capital" : "New Delhi" }
{ "name" : "Indonesia", "capital" : "Jakarta" }
{ "name" : "Iran", "capital" : "Tehran" }
{ "name" : "Japan", "capital" : "Imperial Tokyo" }
{ "name" : "Mexico", "capital" : "Mexico City" }
{ "name" : "Nigeria", "capital" : "Abuja" }
{ "name" : "Pakistan", "capital" : "Islamabad" }
{ "name" : "Philippines", "capital" : "Manila" }
{ "name" : "Russia", "capital" : "Moscow" }
{ "name" : "Turkey", "capital" : "Ankara" }
{ "name" : "United States", "capital" : "Washington, D.C." }
{ "name" : "Vietnam", "capital" : "Hanoi" }
```

Correct answer:

```
{ "name" : "Bangladesh", "capital" : "Dhaka" }
{ "name" : "Brazil", "capital" : "Brasília" }
{ "name" : "China", "capital" : "Beijing" }
{ "name" : "Egypt", "capital" : "Cairo" }
{ "name" : "Germany", "capital" : "Berlin" }
{ "name" : "Ethiopia", "capital" : "Addis Ababa" }
{ "name" : "Iran", "capital" : "Tehran" }
{ "name" : "Indonesia", "capital" : "Jakarta" }
{ "name" : "Japan", "capital" : "Imperial Tokyo" }
{ "name" : "India", "capital" : "New Delhi" }
{ "name" : "Mexico", "capital" : "Mexico City" }
{ "name" : "Pakistan", "capital" : "Islamabad" }
{ "name" : "Nigeria", "capital" : "Abuja" }
{ "name" : "Philippines", "capital" : "Manila" }
{ "name" : "Russia", "capital" : "Moscow" }
{ "name" : "Turkey", "capital" : "Ankara" }
{ "name" : "United States", "capital" : "Washington, D.C." }
{ "name" : "Vietnam", "capital" : "Hanoi" }
```

Using $or

7.

Find the countries that have a population that is over 200 million or less than 20,000.

```
db.world.find({
  $or: [{
    population: {
      $gt: 200000000
    }
  }, {
    population: {
      $lt: 20000
    }
  }]
}, {
  name: 1,
  population: 1,
  _id: 0
});
```

Run NoSQL | Format Answer | Restore Default | Toggle Font Size

Incorrect

Your answer:

```
{ "name" : "Brazil", "population" : 202794000 }
{ "name" : "China", "population" : 1365370000 }
{ "name" : "India", "population" : 1246160000 }
{ "name" : "Indonesia", "population" : 252164800 }
{ "name" : "Nauru", "population" : 9945 }
{ "name" : "Tuvalu", "population" : 11323 }
{ "name" : "United States", "population" : 318320000 }
{ "name" : "Vatican City", "population" : 839 }
```

Correct answer:

```
{ "name" : "Brazil", "population" : 202794000 }
{ "name" : "China", "population" : 1365370000 }
{ "name" : "Indonesia", "population" : 252164800 }
{ "name" : "India", "population" : 1246160000 }
{ "name" : "Nauru", "population" : 9945 }
{ "name" : "Tuvalu", "population" : 11323 }
{ "name" : "United States", "population" : 318320000 }
{ "name" : "Vatican City", "population" : 839 }
```

AGGREGATE Tutorial

$group on continent

1.

The aggregate method allows a `$group` - you must specify the `_id` and you can use aggregating functions such as `$sum` `$min` `$max` `$push`

The sample code shows the total population of each continent.

Show the number of countries in each continent.

```
db.world.aggregate({
  $group: {
    _id: '$continent',
    res: {
      $sum: 1
    }
  }
});
```

Run NoSQL | Format Answer | Restore Default | Toggle Font Size

Your answer:

```
{ "_id" : "North America", "res" : 11 }
{ "_id" : "Oceania", "res" : 14 }
{ "_id" : "Eurasia", "res" : 2 }
{ "_id" : "South America", "res" : 13 }
{ "_id" : "Asia", "res" : 47 }
{ "_id" : "Europe", "res" : 44 }
{ "_id" : "Africa", "res" : 53 }
{ "_id" : "Caribbean", "res" : 11 }
```

Per Capita GDP

2.

Give the `name` and the `per capita GDP` for those countries with a `population` of at least 200 million.

How to calculate per capita GDP

```
db.world.aggregate([
    {$match: {
        population: {$gte: 200000000}
    }},
    {$project: {
        _id: 0,
        name: 1,
        "per capita GDP": {$divide: ['$gdp', '$population']}
    }}
]);
```

Run NoSQL | Format Answer | Restore Default | Toggle Font Size

Incorrect

Your answer:

```
{ "name" : "Brazil", "per capita GDP" : 11115.264751422626 }
{ "name" : "China", "per capita GDP" : 6121.710598592323 }
{ "name" : "India", "per capita GDP" : 1504.7931244783977 }
{ "name" : "Indonesia", "per capita GDP" : 3482.0204881886766 }
{ "name" : "United States", "per capita GDP" : 51032.29454636844 }
```

Correct answer:

```
{ "name" : "Brazil", "per capita GDP" : 11115.264751422626 }
{ "name" : "China", "per capita GDP" : 6121.710598592323 }
{ "name" : "Indonesia", "per capita GDP" : 3482.0204881886766 }
{ "name" : "India", "per capita GDP" : 1504.7931244783977 }
{ "name" : "United States", "per capita GDP" : 51032.29454636844 }
```

Population Density in South America

3.

Give the `name` and the `population density` of all countries in South America.

How to calculate population density

population density is the population divided by the area

Division by 0 error?

Use a `$match`. `{"area":{"$ne":0}}`

```
db.world.aggregate([
    {$match: {continent: 'South America', area: {'$ne':0}}},
    {$project: {
        _id: 0,
        name: 1,
        density: {$divide: ["$population", "$area"]}
    }}
]);
```

Run NoSQL | Format Answer | Restore Default | Toggle Font Size

Your answer:

```
{ "name" : "Argentina", "density" : 15.346532872967918 }
{ "name" : "Bolivia", "density" : 9.127459877787802 }
{ "name" : "Brazil", "density" : 23.81394418142253 }
{ "name" : "Chile", "density" : 23.506087802968384 }
{ "name" : "Colombia", "density" : 41.74476329277564 }
{ "name" : "Ecuador", "density" : 56.97927691346296 }
{ "name" : "Guyana", "density" : 3.651196218989715 }
{ "name" : "Paraguay", "density" : 16.676928447801117 }
{ "name" : "Peru", "density" : 23.712079525931827 }
{ "name" : "Saint Vincent and the Grenadines", "density" : 280.2056555269923 }
{ "name" : "Suriname", "density" : 3.260828958613112 }
{ "name" : "Uruguay", "density" : 18.153020979484516 }
{ "name" : "Venezuela", "density" : 31.585202603538672 }
```

Population Density for "V"

4.

Give the `name` and the `population density` of all countries with name after V in the alphabet.

Note that because Vatican City (with area 0) is in Europe you will get a divide by zero error unless you filter first.

Division by 0 error?

Use a `$match`.

```
{
  $match: {
    area: {
      "$ne": 0
    }
  }
}
```

```
db.world.aggregate([{
  $match: {
    name: {
      $gt: 'V'
    },
    area: {
      '$ne': 0
    }
  }
}, {
  $project: {
    _id: 0,
    name: 1,
    density: {
      $divide: ['$population', '$area']
    }
  }
}]);
```

Run NoSQL | Format Answer | Restore Default | Toggle Font Size

Incorrect

Your answer:

```
{ "name" : "Vanuatu", "density" : 21.712363606530477 }
{ "name" : "Venezuela", "density" : 31.585202603538672 }
{ "name" : "Vietnam", "density" : 270.8503918940135 }
{ "name" : "Yemen", "density" : 47.796457361052184 }
{ "name" : "Zambia", "density" : 19.96156718202739 }
{ "name" : "Zimbabwe", "density" : 33.425476702912555 }
```

Correct answer:

```
{ "name" : "Vanuatu", "density" : 21.712363606530477 }
{ "name" : "Vietnam", "density" : 270.8503918940135 }
{ "name" : "Venezuela", "density" : 31.585202603538672 }
{ "name" : "Zambia", "density" : 19.96156718202739 }
{ "name" : "Yemen", "density" : 47.796457361052184 }
{ "name" : "Zimbabwe", "density" : 33.425476702912555 }
```

Population in millions

5.

Show the `name` and `population` in millions for the countries of the continent **South America**. Divide the population by 1000000 to get population in millions.

```
db.world.aggregate([{
  $match: {
    continent: 'South America'
  }
}, {
  $project: {
    _id: 0,
    name: 1,
    population: {
      $divide: ['$population', 1000000]
    }
  }
}]);
```

Run NoSQL | Format Answer | Restore Default | Toggle Font Size

Your answer:

```
{ "name" : "Argentina", "population" : 42.6695 }
{ "name" : "Bolivia", "population" : 10.027254 }
{ "name" : "Brazil", "population" : 202.794 }
{ "name" : "Chile", "population" : 17.773 }
{ "name" : "Colombia", "population" : 47.662 }
{ "name" : "Ecuador", "population" : 15.7742 }
{ "name" : "Guyana", "population" : 0.784894 }
{ "name" : "Paraguay", "population" : 6.783374 }
{ "name" : "Peru", "population" : 30.475144 }
{ "name" : "Saint Vincent and the Grenadines", "population" : 0.109 }
{ "name" : "Suriname", "population" : 0.534189 }
{ "name" : "Uruguay", "population" : 3.286314 }
{ "name" : "Venezuela", "population" : 28.946101 }
```

Population density

6.

Show the `name` and `population density` for **France**, **Germany**, and **Italy**

```
db.world.aggregate([{
  $match: {
    name: {
      $in: ['France', 'Germany', 'Italy']
    },
    population: {
      $ne: null
    },
    area: {
      $ne: 0
    }
  }
}, {
  $project: {
    _id: 0,
    name: 1,
    'population density': {
      $divide: ['$population', '$area']
    }
  }
}]);
```

Run NoSQL | Format Answer | Restore Default | Toggle Font Size

Your answer:

```
{ "name" : "France", "population density" : 102.86898743364462 }
{ "name" : "Germany", "population density" : 226.0230626634632 }
{ "name" : "Italy", "population density" : 201.71060875567474 }
```

Continents by area

7.

Order the `continents` by `area` from most to least.

```
db.world.aggregate([{
  $group: {
    _id: "$continent",
    area: {
      $sum: "$area"
    }
  }
}, {
  $sort: {
    area: -1
  }
}, {
  $project: {
    _id: 1,
    area: 1
  }
}]);
```

Run NoSQL | Format Answer | Restore Default | Toggle Font Size

Your answer:

```
{ "_id" : "Asia", "area" : 30156669 }
{ "_id" : "Africa", "area" : 26378949 }
{ "_id" : "North America", "area" : 22298259 }
{ "_id" : "South America", "area" : 17738064 }
{ "_id" : "Eurasia", "area" : 17154985 }
{ "_id" : "Europe", "area" : 8672786 }
{ "_id" : "Oceania", "area" : 8489775 }
{ "_id" : "Caribbean", "area" : 205009 }
```

Big Continents

8.

Show the only two continents with total area greater than 25000000 and then sort from largest to smallest.

```
db.world.aggregate([{
  $group: {
    _id: '$continent',
    area: {
      $sum: '$area'
    }
  }
}, {
  $match: {
    area: {
      $gt: 25000000
    }
  }
}, {
  $project: {
    _id: 1,
    area: 1
  }
}]);
```

Run NoSQL | Format Answer | Restore Default | Toggle Font Size

Your answer:

```
{ "_id" : "Asia", "area" : 30156669 }
{ "_id" : "Africa", "area" : 26378949 }
```

First and last country by continent

9.

For each continent show the first and last country alphabetically like this:

```
{ "_id" : "Africa", "from" : "Algeria", "to" : "Zimbabwe" }
{ "_id" : "Asia", "from" : "Afghanistan", "to" : "Yemen" }
{ "_id" : "Caribbean", "from" : "Antigua and Barbuda", "to" : "Trinidad and Tobago" }
{ "_id" : "Eurasia", "from" : "Armenia", "to" : "Russia" }
{ "_id" : "Europe", "from" : "Albania", "to" : "Vatican City" }
{ "_id" : "North America", "from" : "Belize", "to" : "United States" }
{ "_id" : "Oceania", "from" : "Australia", "to" : "Vanuatu" }
{ "_id" : "South America", "from" : "Argentina", "to" : "Venezuela" }
```

```
db.world.aggregate([{
  $group: {
    _id: "$continent",
    small: {
      $min: '$name'
    },
    big: {
      $max: '$name'
    }
  }
}, {
  $sort: {
    _id: 1
  }
}, {
  $project: {
    _id: 1,
    'from': '$small',
    'to': '$big'
  }
}]);
```

Run NoSQL | Format Answer | Restore Default | Toggle Font Size

Your answer:

```
{ "_id" : "Africa", "from" : "Algeria", "to" : "Zimbabwe" }
{ "_id" : "Asia", "from" : "Afghanistan", "to" : "Yemen" }
{ "_id" : "Caribbean", "from" : "Antigua and Barbuda", "to" : "Trinidad and Tobago" }
{ "_id" : "Eurasia", "from" : "Armenia", "to" : "Russia" }
{ "_id" : "Europe", "from" : "Albania", "to" : "Vatican City" }
{ "_id" : "North America", "from" : "Belize", "to" : "United States" }
{ "_id" : "Oceania", "from" : "Australia", "to" : "Vanuatu" }
{ "_id" : "South America", "from" : "Argentina", "to" : "Venezuela" }
```

Countries beginning with...

10.

Group countries according to the first letter of the name. As shown. Only give "U" through to "Z".

You will need to use the **$substr** function and the **$push** aggregate function.

```
{ "_id" : "U", "list" : [ "Uganda", "Ukraine", "United Arab Emirates", "United Kingdom", "United States", "Uruguay", "Uzbekistan" ] }
{ "_id" : "V", "list" : [ "Vanuatu", "Vatican City", "Venezuela", "Vietnam" ] }
{ "_id" : "Y", "list" : [ "Yemen" ] }
{ "_id" : "Z", "list" : [ "Zambia", "Zimbabwe" ] }
```

```
db.world.aggregate([{
  $group: {
    _id: {
      $substr: ['$name', 0, 1]
    },
    list: {
      $push: '$name'
    }
  }
}, {
  $match: {
    _id: {
      $gte: 'U'
    }
  }
}, {
  $sort: {
    _id: 1,
    list: 1
  }
}, {
  $project: {
    _id: 1,
    list: 1
  }
}]);
```

Run NoSQL | Format Answer | Restore Default | Toggle Font Size

Incorrect

Your answer:

```
{ "_id" : "U", "list" : [ "Uganda", "Ukraine", "United Arab Emirates", "United Kingdom", "United States", "Uruguay", "Uzbekistan" ] }
{ "_id" : "V", "list" : [ "Vanuatu", "Vatican City", "Venezuela", "Vietnam" ] }
{ "_id" : "Y", "list" : [ "Yemen" ] }
{ "_id" : "Z", "list" : [ "Zambia", "Zimbabwe" ] }
```

Correct answer:

```
{ "_id" : "U", "list" : [ "Uganda", "Ukraine", "United States", "Uruguay", "United Arab Emirates", "Uzbekistan", "United Kingdom" ] }
{ "_id" : "V", "list" : [ "Vanuatu", "Vietnam", "Venezuela", "Vatican City" ] }
{ "_id" : "Y", "list" : [ "Yemen" ] }
{ "_id" : "Z", "list" : [ "Zambia", "Zimbabwe" ] }
```

AGGREGATE Movies Tutorial

Tom Hanks

2.

You can use the match operator on listed items like cast. As there is an index on cast these queries operate quickly and do not have to scan the entire collection.

Show the title and year of the 10 most recent Tom Hanks movies - show the most recent first.

```
db.movies.aggregate([{
  $match: {
    cast: 'Tom Hanks'
  }
}, {
  $project: {
    _id: 0,
    title: 1,
    yr: 1
  }
}, {
  $sort: {
    'yr': -1
  }
}, {
  $limit: 10
}]).pretty();
```

Run NoSQL | Format Answer | Restore Default | Toggle Font Size

Your answer:

```
{ "title" : "Larry Crowne", "yr" : 2011 }
{ "title" : "Extremely Loud and Incredibly Close", "yr" : 2011 }
{ "title" : "Toy Story 3", "yr" : 2010 }
{ "title" : "Angels & Demons", "yr" : 2009 }
{ "title" : "The Great Buck Howard", "yr" : 2008 }
{ "title" : "Charlie Wilson's War", "yr" : 2007 }
{ "title" : "The Da Vinci Code", "yr" : 2006 }
{ "title" : "The Terminal", "yr" : 2004 }
{ "title" : "The Polar Express", "yr" : 2004 }
{ "title" : "The Ladykillers", "yr" : 2004 }
```