NoSql & MongoDB

1.) Relational Model
- Because this database contains only the student's name, id, and project, which is easier to structure with tables and rows.

2.) MongoDB
- Because this database contains a lot of detailed information which can be easier to handle within JSON format.

3.) MongoDB
- Each reading includes 10 labeled values, so it will be more suitable using JSON.

4.) Gaming:

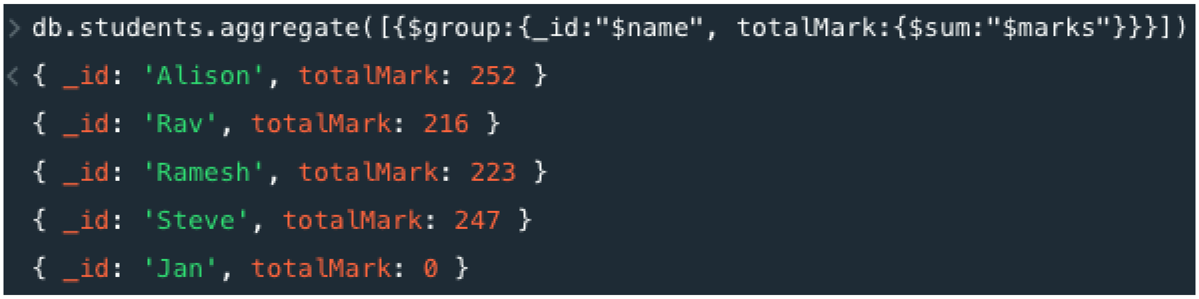
Character	
id	number
name	string
level	number
maxHealth	number
attack	number
defence	number

skill	
id	number
name	string
attack	number

weapon	
id	number
name	string
attack	number
defence	number

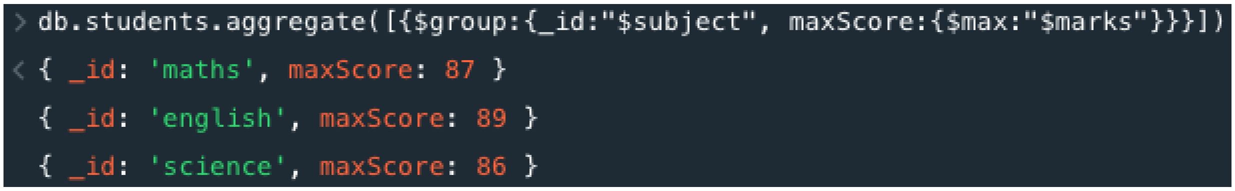
5.)

- Find the total marks for each student across all subjects.



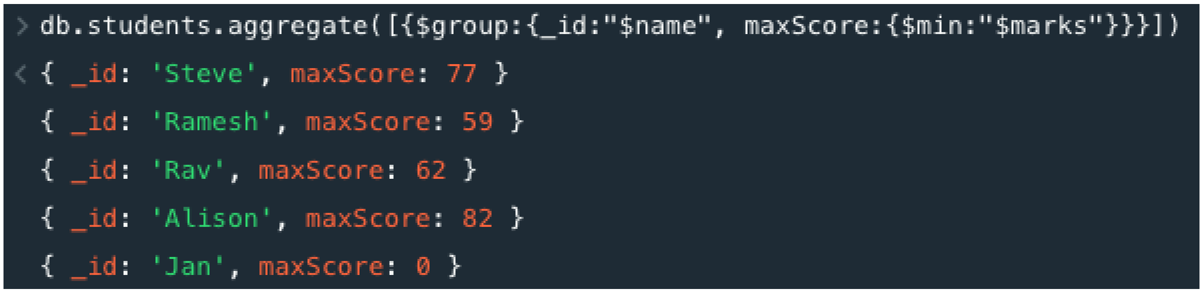
```
> db.students.aggregate([{$group:{_id:"$name", totalMark:{$sum:"$marks"}}}])
< { _id: 'Alison', totalMark: 252 }
  { _id: 'Rav', totalMark: 216 }
  { _id: 'Ramesh', totalMark: 223 }
  { _id: 'Steve', totalMark: 247 }
  { _id: 'Jan', totalMark: 0 }
```

- Find the maximum marks scored in each subject.



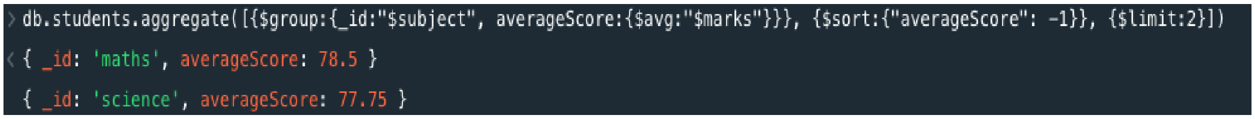
```
> db.students.aggregate([{$group:{_id:"$subject", maxScore:{$max:"$marks"}}}])
< { _id: 'maths', maxScore: 87 }
  { _id: 'english', maxScore: 89 }
  { _id: 'science', maxScore: 86 }
```

- Find the minimum marks scored by each student.



```
> db.students.aggregate([{$group:{_id:"$name", maxScore:{$min:"$marks"}}}])
< { _id: 'Steve', maxScore: 77 }
  { _id: 'Ramesh', maxScore: 59 }
  { _id: 'Rav', maxScore: 62 }
  { _id: 'Alison', maxScore: 82 }
  { _id: 'Jan', maxScore: 0 }
```

- Find the top two subjects based on average marks.



```
db.students.aggregate([{$group:{_id:"$subject", averageScore:{$avg:"$marks"}}}, {$sort:{"averageScore": -1}}, {$limit:2}])
{ _id: 'maths', averageScore: 78.5 }
{ _id: 'science', averageScore: 77.75 }
```