

Homework#NoSQL&MongoDB

- 1) Relational model, because the student information and class project have a fixed structure (name, ID, two students, and a project title). It will be easy if it works with an oriented table and row.
- 2) Relational model, because this one still has a fixed structure. For the optional additional information we can set it as the default null.
- 3) This one I will choose MongoDB. We can group a set of sensor measurements in one field (like a column in relational model). Because maybe we can't define how many sensors that we contained (it's a set of sensors). Maybe one or more. So, we should use MongoDB and collect it as unstructured data.
- 4) I choose d. Finance.

Data schema:

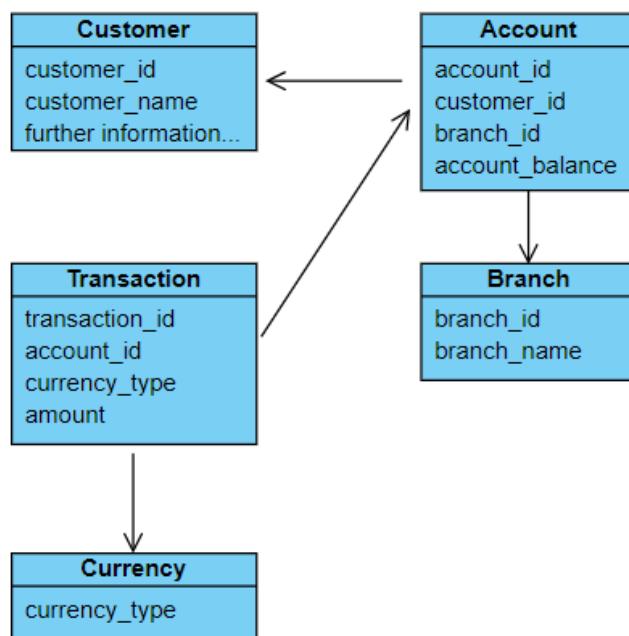
Customer - Customer information.

Account - Account information.

Currency - Each currency.

Transaction - Record the transaction history.

Branch - Branch (สาขา) of Bank.



5)

Create and initialize information:

```
use mongodbHW
db.Q5.insertMany( [
    {"name":"Ramesh","subject":"maths","marks":87},
    {"name":"Ramesh","subject":"english","marks":59},
    {"name":"Ramesh","subject":"science","marks":77},
    {"name":"Rav","subject":"maths","marks":62},
    {"name":"Rav","subject":"english","marks":83},
    {"name":"Rav","subject":"science","marks":71},
    {"name":"Alison","subject":"maths","marks":84},
    {"name":"Alison","subject":"english","marks":82},
    {"name":"Alison","subject":"science","marks":86},
    {"name":"Steve","subject":"maths","marks":81},
    {"name":"Steve","subject":"english","marks":89},
    {"name":"Steve","subject":"science","marks":77},
    {"name":"Jan","subject":"english","marks":0,"reason":"absent"}
])
```

*** If the database or collection did not exist. It will create a new one.

Query:

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

```
db.Q5.aggregate([{$group:{_id:"$name","total":{$sum:"$marks"}}}])
> db.Q5.aggregate([{$group:{_id:"$name", "total":{$sum:"$marks"} }}])
< { _id: 'Jan', total: 0 }
  { _id: 'Rav', total: 216 }
  { _id: 'Steve', total: 247 }
  { _id: 'Alison', total: 252 }
  { _id: 'Ramesh', total: 223 }
```

- Find the maximum marks scored in each subject:

```
db.Q5.aggregate([{$group:{_id:"$subject","max":{$max:"$marks"}}}])
> db.Q5.aggregate([{$group:{_id:"$subject","max":{$max:"$marks"}}}])
< { _id: 'english', max: 89 }
  { _id: 'science', max: 86 }
  { _id: 'maths', max: 87 }
```

- Find the minimum marks scored by each student:

```
db.Q5.aggregate([{$group:{_id:"$name","min":{$min:"$marks"}}}])
> db.Q5.aggregate([{$group:{_id:"$name","min":{$min:"$marks"}}}])
< { _id: 'Jan', min: 0 }
  { _id: 'Rav', min: 62 }
  { _id: 'Steve', min: 77 }
  { _id: 'Alison', min: 82 }
  { _id: 'Ramesh', min: 59 }
  .
```

- Find the top two subjects based on average marks:

```
db.Q5.aggregate([{$group:{_id:"$subject","avg":{$avg:"$marks"}}}])
> db.Q5.aggregate([{$group:{_id:"$subject","avg":{$avg:"$marks"}}}])
< { _id: 'maths', avg: 78.5 }
  { _id: 'english', avg: 62.6 }
  { _id: 'science', avg: 77.75 }
```

```
db.Q5.aggregate([{$group:{_id:"$subject","avg":{$avg:"$marks"}}, {$sort:{avg:-1}}})
> db.Q5.aggregate([{$group:{_id:"$subject","avg":{$avg:"$marks"}}, {$sort:{avg:-1}}}])
< { _id: 'maths', avg: 78.5 }
  { _id: 'science', avg: 77.75 }
  { _id: 'english', avg: 62.6 }
```

```
db.Q5.aggregate([{$group:{_id:"$subject","avg":{$avg:"$marks"}},  
{$sort:{avg:-1}}, {$limit:2}])  
> db.Q5.aggregate([{$group:{_id:"$subject","avg":{$avg:"$marks"}}, {$sort:{avg:-1}}, {$limit:2}}])  
< { _id: 'maths', avg: 78.5 }  
{ _id: 'science', avg: 77.75 }
```