REPORT

I Have completed my SQL lab with help of notes provided during the regular online classes as well as reference book shared in class that is “book **BeginningSQLqueries** by Clare Churcher”.

I got required the output (correct format) after rectifying silly error in almost each queries. I Refer website such as https://stackoverflow.com/, https://www.geeksforgeeks.org/ https://www.w3schools.com/ , to get know different ways of writing queies.

Question: 1

```
> db.createCollection("students")
{"ok" : 1}
```

Question: 2

```
> db.students.insertMany([{"_id": "arun", "name": "Arun Kumar",
"year": "1992", "courses": ["Java", "php"]}])

> db.students.insertMany([{"_id": "sam", "name": "sam peter",
"year": "1995", "courses": ["php", "python",
"Java"]}])

> db.students.insertMany([{"_id": "anna", "name": "anna eva",
"year": "1997", "courses": ["java"]}])

> db.students.insertMany([{"_id": "rex", "name": "rex
Samuel", "year": "1998", "courses": ["python"]}])

> db.students.insertMany([{"_id": "olivia", "name": "olivia
cathy", "year": "2006"}])
```

Lab9. Student Information System Design using MongoDB PART-I

Question1. Create a new collection **students**

Question2. Insert the following students into your **students** collection

```
{       _id: "arun",
        name: "arun kumar", year: 1992,
        courses: [ "java", "php" ]
}
{       _id: "sam",
        name: "sam peter", year: 1995,
        courses: [ "php", "python", "java" ]
}
{       _id: "anna",
        name: "anna eva", year: 1997,
        courses: [ "java" ]
}
{       _id: "rex",
        name: "rex samuel", year: 1988,
        courses: [ "python" ]
}
{       _id: "olivia",
        name: "olivia cathy", year: 2006
}
```

1) db.students.find()

> db.students.find()

{"_id":"arun","name":"arunkumar","year":1992,"courses":["java","php"]}

{"_id":"sam","name":"sam peter","year":1995,"courses":["php","python","java"]}

{"_id":"anna","name":"anna eva","year":1997,"courses":["java"]}

{"_id":"rex","name":"rex samuel","year":1988,"courses":["python"]}

{"_id":"olivia","name":"olivia cathy","year":2006}

2) db.students.find({}):

> db.students.find({})

{"_id":"arun","name":"arun kumar","year":1992,"courses":["java","php"]}

{"_id":"sam","name":"sam peter","year":1995,"courses":["php","python","java"]}

{"_id":"anna","name":"anna eva","year":1997,"courses":["java"]}

{"_id":"rex","name":"rex samuel","year":1998,"courses":["python"]}

{"_id":"olivia","name":"olivia cathy","year":2006}

3) db.students.find({_id:"arun"}):

> db.students.find({_id:"arun"})

{"_id":"arun","name":"arun kumar","year":1992,"courses":["java","php"]}

4) db.students.find({name:"arun kumar",year:1992}):

> db.students.find({name:"arun kumar",year:1992})

{"_id":"arun","name":"arun kumar","year":1992,"courses":["java","php"]}

Question3. Execute and explain the meaning of the following queries

db.students.find()
db.students.find({ })
db.students.find({ _id: "arun" })
db.students.find({ name: "arun kumar", year: 1992 })
db.students.find({ year: { $gte: 1990, $lte: 2000 } }) {"_id": "arun", "name": "arun kumar", "year": 1992, "courses": ["java", "php"]} {"_id": "sam", "name": "sam peter", "year": 1995, "courses": ["php", "python", "java"]} {"_id": "anna", "name": "anna eva", "year": 1997, "courses": ["java"]}
db.students.find({ courses: { $exists: true } }) {"_id": "arun", "name": "arun kumar", "year": 1992, "courses": ["java", "php"]} {"_id": "sam", "name": "sam peter", "year": 1995, "courses": ["php", "python", "java"]} {"_id": "anna", "name": "anna eva", "year": 1997, "courses": ["java"]}
db.students.find({ courses: " php " }) {"_id": "arun", "name": "arun kumar", "year": 1992, "courses": ["java", "php"]} {"_id": "sam", "name": "sam peter", "year": 1995, "courses": ["php", "python", "java"]}
db.students.find({ courses: { $in: ["php", "oracle"] } }) {"_id": "arun", "name": "arun kumar", "year": 1992, "courses": ["java", "php"]} {"_id": "sam", "name": "sam peter", "year": 1995, "courses": ["php", "python", "java"]}
db.students.find({ courses: { $all: ["php", "oracle"] } })

2) db.students.find({rating:{$not:{$gte:3}}}):

{"_id":"arun", "name":"arun kumar", "year":1992, "courses":["java", "php"]}

{"_id":"Sam", "name":"Sam peter", "year":1995, "courses":["php", "python", "java"]}

{"_id":"anna", "name":"anna eva", "year":1997, "courses":["Java"]}

{"_id":"rex", "name":"rex Samuel", "year":1988, "courses":["python"]}

{"_id":"olivia", "name":"olivia cathy", "year":2006}

6) db.students.find().sort({year:1, name:-1}):

{"_id":"rex", "name":"rex samuel", "year":1988, "courses":["python"]}

{"_id":"arun", "name":"arunkumar", "year":1992, "courses":["java", "php"]}

{"_id":"Sam", "name":"Sam peter", "year":1995, "courses":["php", "python", "java"]}

{"_id":"anna", "name":"anna eva", "year":1997, "courses":["java"]}

{"_id":"olivia", "name":"olivia cathy", "year":2006}

Question4. Execute and explain the meaning of the following queries

db.students.find({ $or: [{ year: 1992 }, { rating: { $gte: 3 } }] })

{"_id": "arun", "name": "arunkumar", "year": 1992, "courses": ["java", "php"]}

db.students.find({ rating: { $not: { $gte: 3 } } })

db.students.find({ }, { name: 1, year: 1 })

{"_id": "arun", "name": "arunkumar", "year": 1992}
{"_id": "sam", "name": "sam peter", "year": 1995}
{"_id": "anna", "name": "anna eva", "year": 1997}
{"_id": "rex", "name": "rex samuel", "year": 1988}

db.students.find({ }, { courses: 0, _id: 0 })

{"name": "arunkumar", "year": 1992}
{"name": "sam peter", "year": 1995}
{"name": "anna eva", "year": 1997}
{"name": "rex samuel", "year": 1988, "course": ["python"]}

db.students.find({ }, { name: 1, courses: { $slice: 2 }, _id: 0 })

{"name": "arunkumar", "courses": ["java", "php"]}
{"name": "sam peter", "courses": ["php", "python"]}
{"name": "anna eva", "courses": ["java"]}
{"name": "rex samuel"}

db.students.find().sort({ year: 1, name: -1 })

db.students.find().sort({ name: 1 }).skip(1).limit(2)

{"_id": "arun", "name": "arunkumar", "year": 1992, "courses": ["java", "php"]}
{"_id": "olivia", "name": "olivia cathy", "year": 2006}

db.students.find().sort({ name: 1 }).limit(2).skip(1)

{"_id": "arun", "name": "arunkumar", "year": 1992, "courses": ["java", "php"]}
{"_id": "olivia", "name": "olivia cathy", "year": 2006}

LAB9: Student Information System Design using MongoDB PART-I

Question1. Create a new collection students.

```
> db.createCollection("students")
{ "ok" : 1 }
```

Question2. Insert the following students into your student collection.

```
> db.students.insert({_id:"arun",name:"arun
kumar",year:1992,courses:["java","php"]})
WriteResult({ "nInserted" : 1 })
> db.students.insert({_id:"sam",name:"sam
peter",year:1995,courses:["php","python","java"]})
WriteResult({ "nInserted" : 1 })
> db.students.insert({_id:"anna",name:"anna
eva",year:1997,courses:["java"]})
WriteResult({ "nInserted" : 1 })
> db.students.insert({_id:"rex",name:"rex
samuel",year:1988,courses:["python"]})
WriteResult({ "nInserted" : 1 })
> db.students.insert({_id:"olivia",name:"olivia
cathy",year:2006})
WriteResult({ "nInserted" : 1 })
```

Question3. Execute and explain the meaning of the following Queries.

db.students.find():

```
> db.students.find()
{ "_id" : "arun", "name" : "arun kumar", "year" : 1992,
"courses" : [ "java", "php" ] }
{ "_id" : "sam", "name" : "sam peter", "year" : 1995, "courses" :
[ "php", "python", "java" ] }
{ "_id" : "anna", "name" : "anna eva", "year" : 1997, "courses" :
[ "java" ] }
{ "_id" : "rex", "name" : "rex samuel", "year" : 1988, "courses" :
[ "python" ] }
{ "_id" : "olivia", "name" : "olivia cathy", "year" : 2006 }
```

db.students.find({ }):

```
> db.students.find({})
{ "_id" : "arun", "name" : "arun kumar", "year" : 1992,
"courses" : [ "java", "php" ] }
```

{ "_id" : "sam", "name" : "sam peter", "year" : 1995, "courses" :
["php", "python", "java"] }
{ "_id" : "anna", "name" : "anna eva", "year" : 1997, "courses" :
["java"] }
{ "_id" : "rex", "name" : "rex samuel", "year" : 1988, "courses" :
["python"] }
{ "_id" : "olivia", "name" : "olivia cathy", "year" : 2006 }

db.students.find({ _id: "arun" }):
> db.students.find({_id:"arun"})
{ "_id" : "arun", "name" : "arun kumar", "year" : 1992,
"courses" : ["java", "php"] }

db.students.find({ name: "arun kumar", year: 1992 }):
> db.students.find({name:"arun kumar",year:1992})
{ "_id" : "arun", "name" : "arun kumar", "year" : 1992,
"courses" : ["java", "php"] }

db.students.find({ year: { $gte: 1990, $lte: 2000 } }):
> db.students.find({year:{$gte:1990,$lte:2000}})
{ "_id" : "arun", "name" : "arun kumar", "year" : 1992,
"courses" : ["java", "php"] }
{ "_id" : "sam", "name" : "sam peter", "year" : 1995, "courses" :
["php", "python", "java"] }
{ "_id" : "anna", "name" : "anna eva", "year" : 1997, "courses" :
["java"] }

db.students.find({ courses: { $exists: true } }):
> db.students.find({courses:{$exists:true}})
{ "_id" : "arun", "name" : "arun kumar", "year" : 1992,
"courses" : ["java", "php"] }
{ "_id" : "sam", "name" : "sam peter", "year" : 1995, "courses" :
["php", "python", "java"] }
{ "_id" : "anna", "name" : "anna eva", "year" : 1997, "courses" :
["java"] }

db.students.find({ courses: " php " }):
> db.students.find({courses:"php"})
{ "_id" : "arun", "name" : "arun kumar", "year" : 1992,
"courses" : ["java", "php"] }
{ "_id" : "sam", "name" : "sam peter", "year" : 1995, "courses" :
["php", "python", "java"] }

db.students.find({ courses: { $in: ["php", "oracle"] } }):
> db.students.find({courses:{$in:["php","oracle"]}})
{ "_id" : "arun", "name" : "arun kumar", "year" : 1992,
"courses" : ["java", "php"] }

{ "_id" : "sam", "name" : "sam peter", "year" : 1995, "courses" :
["php", "python", "java"] }

db.students.find({ courses: { $all: ["php", "oracle"] } }):
> db.students.find({courses:{$all:["php","oracle"]}})
>

Question4. Execute and explain the meaning of the following queries.

db.students.find({ $or: [{ year: 1992 }, { rating: { $gte: 3 } }] }):
> db.students.find({$or:[{year:1992},{rating:{$gte:3}}]})
{ "_id" : "arun", "name" : "arun kumar", "year" : 1992,
"courses" : ["java", "php"] }

db.students.find({ rating: { $not: { $gte: 3 } } }):
> db.students.find({rating:{$not:{$gte:3}}})
{ "_id" : "arun", "name" : "arun kumar", "year" : 1992,
"courses" : ["java", "php"] }
{ "_id" : "sam", "name" : "sam peter", "year" : 1995, "courses" :
["php", "python", "java"] }
{ "_id" : "anna", "name" : "anna eva", "year" : 1997, "courses" :
["java"] }
{ "_id" : "rex", "name" : "rex samuel", "year" : 1988, "courses" :
["python"] }
{ "_id" : "olivia", "name" : "olivia cathy", "year" : 2006 }

db.students.find({ }, { name: 1, year: 1 }):
> db.students.find({},{name:1,year:1})
{ "_id" : "arun", "name" : "arun kumar", "year" : 1992 }
{ "_id" : "sam", "name" : "sam peter", "year" : 1995 }
{ "_id" : "anna", "name" : "anna eva", "year" : 1997 }
{ "_id" : "rex", "name" : "rex samuel", "year" : 1988 }
{ "_id" : "olivia", "name" : "olivia cathy", "year" : 2006 }

db.students.find({ }, { courses: 0, _id: 0 }):
> db.students.find({},{courses:0,_id:0})
{ "name" : "arun kumar", "year" : 1992 }
{ "name" : "sam peter", "year" : 1995 }
{ "name" : "anna eva", "year" : 1997 }
{ "name" : "rex samuel", "year" : 1988, "courses" :["python"] }
{ "name" : "olivia cathy", "year" : 2006 }

db.students.find({ }, { name: 1, courses: { $slice: 2 }, _id:0 }):
> db.students.find({},{name:1,courses:{$slice:2},_id:0})
{ "name" : "arun kumar", "courses" : ["java", "php"] }
{ "name" : "sam peter", "courses" : ["php", "python"] }

{ "name" : "anna eva", "courses" : ["java"] }
{ "name" : "rex samuel" }
{ "name" : "olivia cathy" }

db.students.find().sort({ year: 1, name: -1 }):

```
> db.students.find().sort({year:1,name:-1})
{ "_id" : "rex", "name" : "rex samuel", "year" : 1988, "courses" :
[ "python" ] }
{ "_id" : "arun", "name" : "arun kumar", "year" : 1992,
"courses" : [ "java", "php" ] }
{ "_id" : "sam", "name" : "sam peter", "year" : 1995, "courses" :
[ "php", "python", "java" ] }
{ "_id" : "anna", "name" : "anna eva", "year" : 1997, "courses" :
[ "java" ] }
{ "_id" : "olivia", "name" : "olivia cathy", "year" : 2006 }
```

db.students.find().sort({ name: 1 }).skip(1).limit(2):

```
> db.students.find().sort({name:1}).skip(1).limit(2)
{ "_id" : "arun", "name" : "arun kumar", "year" : 1992,
"courses" : [ "java", "php" ] }
{ "_id" : "olivia", "name" : "olivia cathy", "year" : 2006 }
```

db.students.find().sort({ name: 1 }).limit(2).skip(1):

```
> db.students.find().sort({name:1}).limit(2).skip(1)
{ "_id" : "arun", "name" : "arun kumar", "year" : 1992,
"courses" : [ "java", "php" ] }
{ "_id" : "olivia", "name" : "olivia cathy", "year" : 2006 }
```

REPORT

I Have completed my SQL lab with help of notes provided during the regular online classes as well as reference book shared in class that is “book **BeginningSQLqueries** by Clare Churcher”.

I got required the output (correct format) after rectifying silly error in almost each queries. I Refer website such as https://stackoverflow.com/, https://www.geeksforgeeks.org/ https://www.w3schools.com/ , to get know different ways of writing queies.

Lab10. Student Information System Design using MongoDB PART-II

Question1. Execute the following queries in your **students** collection

```
db.students.save({
        _id: "arun",
        name: { first: "Arun", last:"Kumar" }, year: 1992,
        courses: [ "java", "php", "mongodb"]
});

db.students.save({
        _id: "sam",
        name: { first: "Sam", last:"Peter" }, year: 1995,
        courses: [ "php", "python", "java" ]
});

db.students.save({
        _id: "anna",
        name: {first: "Anna", last:"Eva" }, year: 1997,
        courses: [ "java" ]
});

db.students.save({
        _id: "rex",
        name: {first: "Rex", last:"Samuel" }, year: 1988,
        courses: [ "python" ]
});

db.students.save({
        _id: "olivia",
        name: { first: "Olivia", last:"Freda" }, year: 2006
});

db.students.save({
  _id: "sylvia",
  name: { first: "Sylvia", last: "Diana" }, year: 2008,
  courses: [ "php"]
});

db.students.save({
  _id: "Benita",
  name: { last: "Benita", first: "Sam" }, year: 1988,
  courses: "php"
```

});

Question2. Execute the following queries in your **courses** collection

```
db.courses.save({
   _id: "java",
   title: { tamil: "Java Programming", en: "Java Programming" },
   year: 2000, rating: 84,
   students: [ "arun", "sam", "anna" ],
   departments: [ "cs", "ca" ], campus: [ "Trichy", "Dubai" ] });

db.courses.save({
   _id: "php",
   title: "PHP Programming",
   year: 2007, professor: { first: "John", last: "Peter" }, rating: 53,
   students: [ "arun", "sam", "sylvia", "divya", "hazel" ],
   departments: [ "cs", "ca" ], campus: [ "Trichy","Singapore" ] });

db.courses.save({
   _id: "python",
   title: { tamil: "Python Programming", en: "Python Programming" },
   year: 2006, professor: { first: "John", last: "Santhosh" }, rating: 76,
   students: [ "rex", "sam", "anna" ],
   departments: "cs", campus: "Trichy" });

db.courses.save({
   _id: "os",
   title: "Operating Systems",
   year: 2003, professor: { last: "Titus", first: "Antony" }, rating: 81,
   students: [ ],
   departments: [ "ds", "ca" ], campus: [ "Trichy", "Sharjah", "Singapore" ] });

db.courses.save({
   _id: "networks",
   title: "Networking Fundamentals",
   year: 2005, professor: { last: "Solomon", first: "Balu" }, rating: 72,
   awards: [
      { type: "National Award", year: 2005 } ] });

db.courses.save({
   _id: "graphics",
   title: "Computer Graphics",
   year: 1996, rating: 86,
   awards: [
      { type: " National Award ", year: 1996 },
      { type: " International Award ", category: "Core CS", year: 2005 } ] });
```

Question3. Express the following MongoDB queries, execute and find answers

1. Find students born in 1995 with first name Sam

```
> db.students.find({"name.first": "sam", year:1995})
{"id": "sam", "name":{"first": "sam", "last": "peter"}, "year":
1995, "courses":["java", "php", "python"]}
```

2. Find courses delivered by John Peter

- Note that the order of fields for first and last names can be arbitrary

```
> db.courses.find({"professor.first": "john", "professor.last":
"peter"})
{"id": "php", title": "php programming", "year": 2007, "professor":
{"first": "john", "last": "peter"}, "rating": 53, "students":
["arun", "sam", "sylia", "divya", "hazel"], "depts":["cs", "ca"],
"campus": ["trichy", "singapore"]}
```

3. Find students with first name *Sam* who take the course *php*

- Return names of these students only

```
> db.students.find({"name.first":"sam", courses: "php"},
{name:1, _id:0})
```

4. Find courses delivered between years *2000* and *2005* such that they have a professor specified

- Return course identifier only
- Order the result by ratings in descending order and then by years in ascending order

```
> db.courses.find({year:{$gte:2000, $lte:2005}}, {_id:1}).
  sort({rating:-1, year:1})
```

5. Find students who studied courses, *java* or *php*

- Return student identifier only
- Propose two different approaches

```
> db.Students.find({courses:{$in:["php","Java"]}},
  {_id:1})
```

6. Find students who studied courses both *java* and *php*

- Return student identifier only
- Propose two different approaches

```
> db.Students.find({courses:{$all:["php", "Java"]}},
  {_id:1})
```

7. Find courses with Tamil title equal to *Python Programming*

- Return course title only
- Note that there are two ways how course titles are defined

```
>db.courses.find({"title.tamil": "Java Programming"}, {title:1,
-id:0})
```

8. Find courses that have a *National* award from *2005*

- Return course identifier and all awards

```
>db.courses.find({"awards.type":"national
awards", year:{$gte:2005}}, {_id:1, "awards.type":1})
```

9. Find courses that are offered in CS and CA departments at the same time or have a rating *80* or more

- Return course identifier and at most 2 campuses

```
>db.courses.find({$or:[{depts:{$all:["CS", "CA"]}},
{rating:{$gte:80}}]}, {_id:1, campus:{$slice:2}})
```

LAB10: STUDENT INFORMATION SYSTEM DESIGN USING MONGODB P-||

Question1.execute the following queries in your Students collection

>db.students.save({_id:"arun",name:{first:"Arun",last:"kumar"},year:1992,
courses:["java","php","mongodb"]});
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })

>db.students.save({_id:"sam",name:{first:"Sam",last:"Peter"},year:1995,
courses:["php","python","java"]});
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })

>db.students.save({_id:"anna",name:{first:"Anna",last:"Eva"},year:1997,
courses:["java"]});
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })

>db.students.save({_id:"rex",name:{first:"Rex",last:"Samuel"},year:1988,
courses:["python"]});
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })

>db.students.save({_id:"olivia",name:{first:"Olivia",last:"Freda"},year:2006});
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })

>db.students.save({_id:"sylvia",name:{first:"Sylvia",last:"Diana"},year:2008,
courses:["php"]});
WriteResult({ "nMatched" : 0, "nUpserted" : 1, "nModified" : 0, "_id" : "sylvia" })

>db.students.save({_id:"Benita",name:{first:"Benita",last:"Sam"},year:1988,
courses:"php"});
WriteResult({ "nMatched" : 0, "nUpserted" : 1, "nModified" : 0, "_id" : "Benita" })

Question2.Execute the following queries in your courses collection

> db.courses.save({_id:"java",title:{tamil:"Java Programming",en:"Java Programming"},year:2000,rating:84,students:["arun","sam","anna"],departments:["cs","ca"],campus:["Trichy","Dubai"]});;

WriteResult({ "nMatched" : 0, "nUpserted" : 1, "nModified" : 0, "_id" : "java" })

> db.courses.save({_id:"php",title:"PHP Programming",year:2007,professor:{first:"John",last:"Peter"},rating:53,students:["arun","sam","sylvia","divya","hazel"],departments:["cs","ca"],campus:["Trichy","Singapore"]});

WriteResult({ "nMatched" : 0, "nUpserted" : 1, "nModified" : 0, "_id" : "php" })

**> db.courses.save({_id:"python",title:{tamil:"Pythonprogramming",
en:"Python Programming"},
year:2006,professor:{first:"John",last:"Santhosh"},rating:76,
students:["rex","sam","anna"],departments:"cs",campus:"Trichy"});**

WriteResult({ "nMatched" : 0, "nUpserted" : 1, "nModified" : 0, "_id" : "python" })

**>db.courses.save({_id:"os",title:"OperatingSystems",year:2003,professor:{last:"Titus",first:"Antony"},rating:81,students:[],departments:
["ds","ca"],campus:["Trichy","Sharjah","Singapore"]});**

WriteResult({ "nMatched" : 0, "nUpserted" : 1, "nModified" : 0, "_id" : "os" })

**> db.courses.save({_id:"networks",title:"NetworkingFundamentals",
year:2005,professor:{last:"Solomon",first:"Balu"},rating:72,awards:[{type:"National Award",year:2005}]});**

WriteResult({ "nMatched" : 0, "nUpserted" : 1, "nModified" : 0, "_id" : "networks" })

**> db.courses.save({_id:"graphics",title:"ComputerGraphics",year:1996,
rating:86,awards:[{type:"NationalAward",year:1996},{type:"National Award",year:1996},{type:"International Award",category:"Core CS",year:2005}]});**

WriteResult({ "nMatched" : 0, "nUpserted" : 1, "nModified" : 0, "_id" : "graphics" })

LAB 10: STUDENT INFORMATION SYSTEM DESIGN USING MONGODB

Question 3: Express the following queries, execute and find answers

1. Find students born in 1995 with first name Sam

```
> db.students.find({"name.first":"sam",year:1995})
```

```
{ "_id" : "sam", "name" : { "first" : "sam", "last" : "peter" }, "year" : 1995,
"courses" : [ "java", "php", "python" ] }
```

2. Find courses delivered by John Peter

```
> db.courses.find({"professor.first":"john","professor.last":"peter"})
```

```
{ "_id" : "PHP", "title" : "PHP programmimg", "year" : 2007, "professor" : {
"first" : "john", "last" : "peter" }, "rating" : 53, "students" : [ "arun", "sam",
"sylia", "divya", "hazel" ], "depts" : [ "cs", "ca" ], "campus" : [ "trichy",
"singapore" ] }
```

3. Find students with first name *Sam* who take the course *php.*
(return name of these students only)

```
> db.students.find({"name.first":"sam",courses:"php"},{name:1,_id:0})
```

```
{ "name" : { "first" : "sam", "last" : "peter" } }
```

4. Find courses delivered between years *2000* and *2005* such that they have a professor specified.
(return course identifier only
Order of result by rating in desc and by years in aesc)

```
> db.courses.find({year:{$gte:2000,$lte:2005}},{_id:1}).sort({rating:-1,year:1})
```

```
{ "_id" : "java" }
```

{ "_id" : "os" }

{ "_id" : "networks" }

5. Find students who studied courses, *java* or *php.* (<u>return student indentifier only)</u>

> db.students.find({courses:{$in:["php","java"]}},{_id:1})

{ "_id" : "arun" }

{ "_id" : "sam" }

{ "_id" : "anna" }

{ "_id" : "sylvia" }

{ "_id" : "benita" }

6. Find students who studied courses both *java* and *php.* (return student identifier only)

> db.students.find({courses:{$all:["php","java"]}},{_id:1})

{ "_id" : "arun" }

{ "_id" : "sam" }

7. Find courses with Tamil title equal to *Python Programming*.
<u>(return course title only)</u>

> db.courses.find({"title.tamil":"java programming"},{title:1,_id:0})

{ "title" : { "tamil" : "java programming", "en" : "java programmimg" } }

8. Find courses that have a *National* award from *2005* (return course identifier and all awards)

```
> db.courses.find({"awards.type":"national
awards",year:{$gte:2005}},{_id:1,"awards.type":1})
```

```
{ "_id" : "networks", "awards" : [ { "type" : "national awards" } ] }
```

9. Find courses that are offered in CS and CA departments at the same time or have a rating *80* or more. (return course identifier and at most 2 campus)

```
> db.courses.find({$or:[{depts:{$all:["cs","ca"]}} ,
{rating:{$gte:80}}]},{_id:1,campus:{$slice:2}})
```

```
{ "_id" : "java", "campus" : [ "trichy", "dubai" ] }
```

```
{ "_id" : "PHP", "campus" : [ "trichy", "singapore" ] }
```

```
{ "_id" : "graphics" }
```

```
{ "_id" : "os", "campus" : [ "trichy", "sharjah"] }
```