EX.NO.12   PERFORM MONGODB QUERY AND PROJECTION OPERATION

Aim:

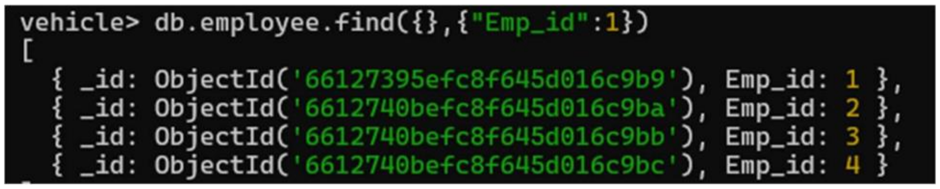
To execute queries and projection operations using MongoDB.

Procedure:

PROJECTION:

MongoDB Projection is used when we want to get the selected fields of the documents rather than all fields.

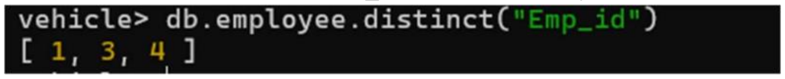
Specific column(s)



```
vehicle> db.employee.find({},{"Emp_id":1})
[
  { _id: ObjectId('66127395efc8f645d016c9b9'), Emp_id: 1 },
  { _id: ObjectId('6612740befc8f645d016c9ba'), Emp_id: 2 },
  { _id: ObjectId('6612740befc8f645d016c9bb'), Emp_id: 3 },
  { _id: ObjectId('6612740befc8f645d016c9bc'), Emp_id: 4 }
```

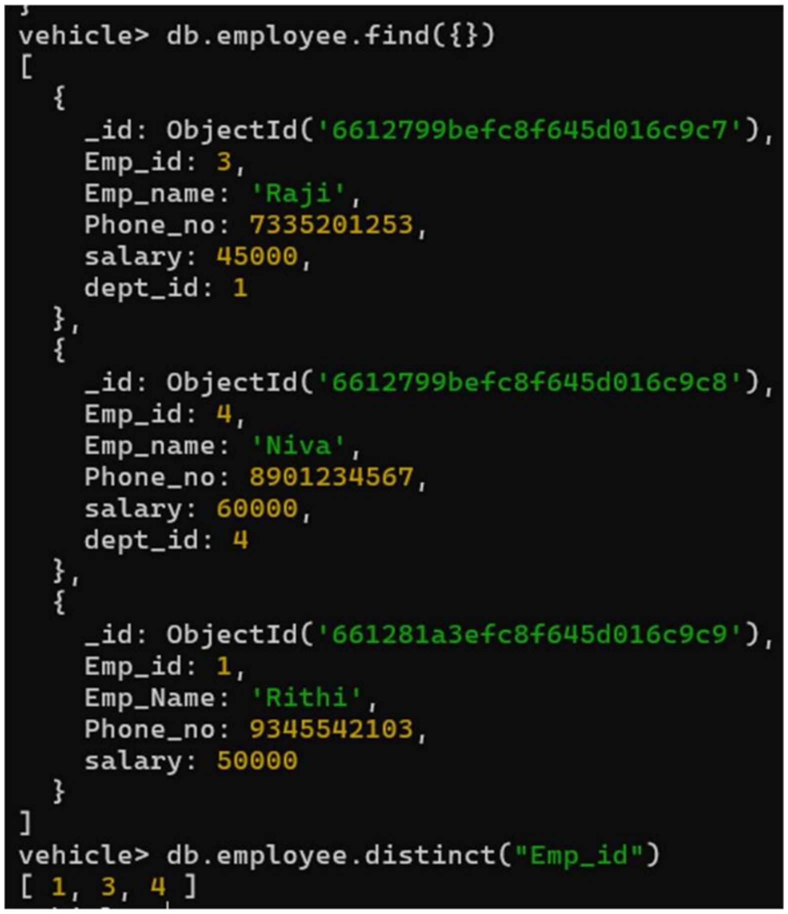
QUERIES:

- SELECT DISTINCT: db.collection_name.distinct(“key”)

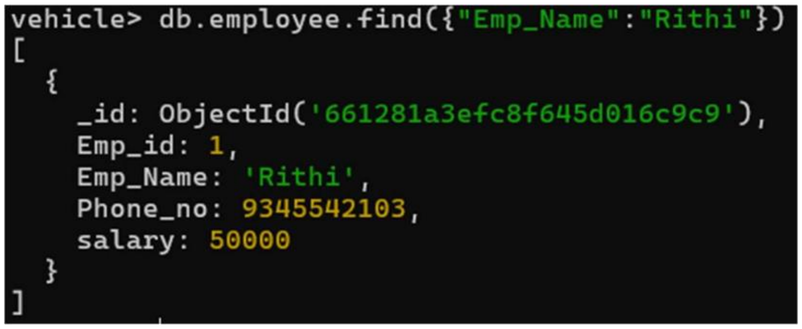


```
vehicle> db.employee.distinct("Emp_id")
[ 1, 3, 4 ]
```

- FIND: (Select in SQL)   All columns

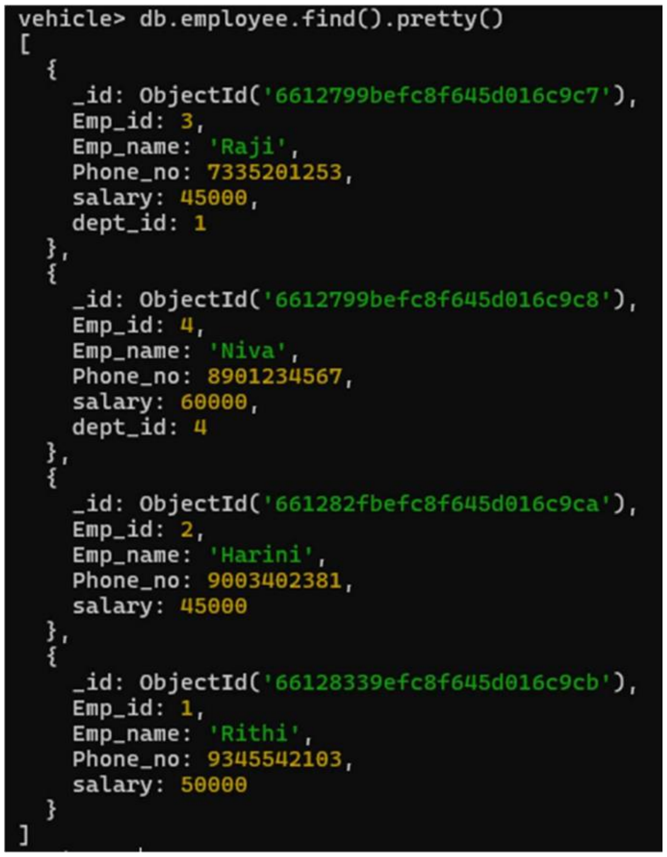


```
vehicle> db.employee.find({})
[
  {
    _id: ObjectId('6612799befc8f645d016c9c7'),
    Emp_id: 3,
    Emp_name: 'Raji',
    Phone_no: 7335201253,
    salary: 45000,
    dept_id: 1
  },
  {
    _id: ObjectId('6612799befc8f645d016c9c8'),
    Emp_id: 4,
    Emp_name: 'Niva',
    Phone_no: 8901234567,
    salary: 60000,
    dept_id: 4
  },
  {
    _id: ObjectId('661281a3efc8f645d016c9c9'),
    Emp_id: 1,
    Emp_Name: 'Rithi',
    Phone_no: 9345542103,
    salary: 50000
  }
]
vehicle> db.employee.distinct("Emp_id")
[ 1, 3, 4 ]
```



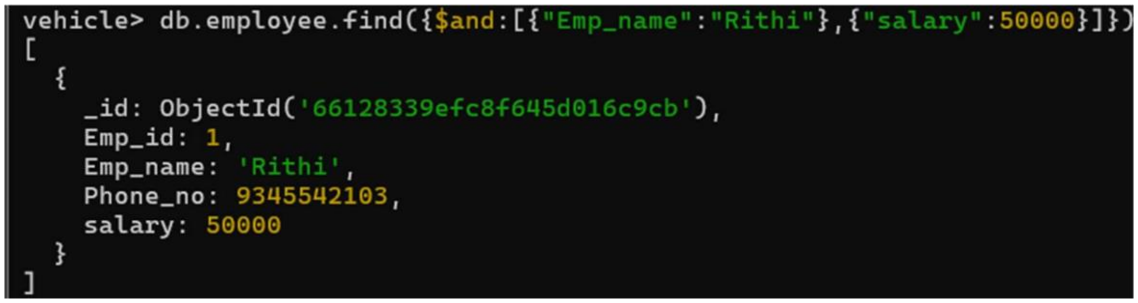
```
vehicle> db.employee.find({"Emp_Name":"Rithi"})
[
  {
    _id: ObjectId('661281a3efc8f645d016c9c9'),
    Emp_id: 1,
    Emp_Name: 'Rithi',
    Phone_no: 9345542103,
    salary: 50000
  }
]
```

To display the results in the formatted way : find().pretty()



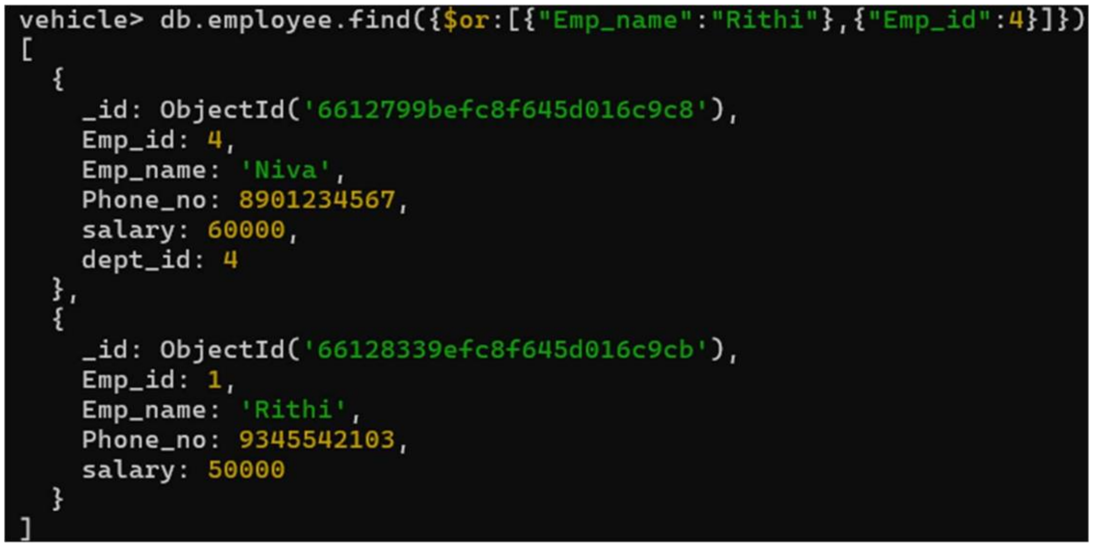
```
vehicle> db.employee.find().pretty()
[
  {
    _id: ObjectId('6612799befc8f645d016c9c7'),
    Emp_id: 3,
    Emp_name: 'Raji',
    Phone_no: 7335201253,
    salary: 45000,
    dept_id: 1
  },
  {
    _id: ObjectId('6612799befc8f645d016c9c8'),
    Emp_id: 4,
    Emp_name: 'Niva',
    Phone_no: 8901234567,
    salary: 60000,
    dept_id: 4
  },
  {
    _id: ObjectId('661282fbefc8f645d016c9ca'),
    Emp_id: 2,
    Emp_name: 'Harini',
    Phone_no: 9003402381,
    salary: 45000
  },
  {
    _id: ObjectId('66128339efc8f645d016c9cb'),
    Emp_id: 1,
    Emp_name: 'Rithi',
    Phone_no: 9345542103,
    salary: 50000
  }
]
```

- CONDITIONAL SELECT: db.collection_name.find({key:value})
    1. AND: db.collection_name.find({ $and: [{key:value},…,{key:value}] })



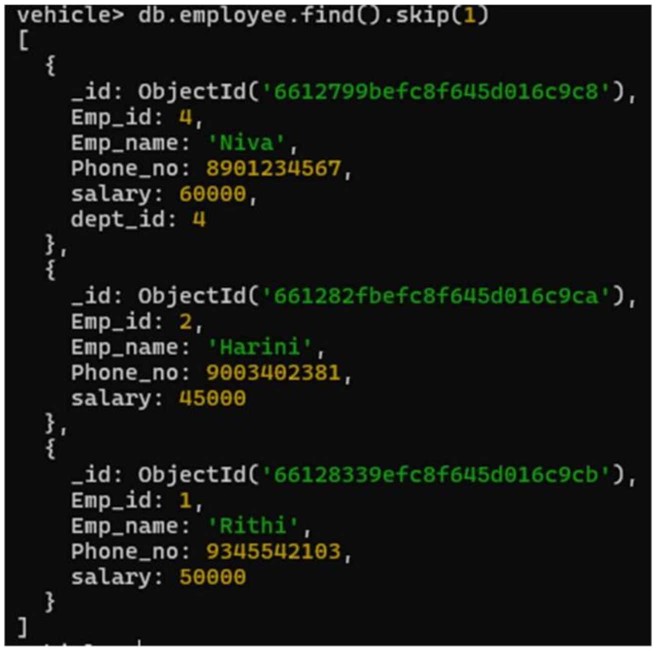
```
vehicle> db.employee.find({$and:[{"Emp_name":"Rithi"},{"salary":50000}]})
[
  {
    _id: ObjectId('66128339efc8f645d016c9cb'),
    Emp_id: 1,
    Emp_name: 'Rithi',
    Phone_no: 9345542103,
    salary: 50000
  }
]
```

    2. OR: db.collection_name.find({ $or: [{key:value},…,{key:value}] })



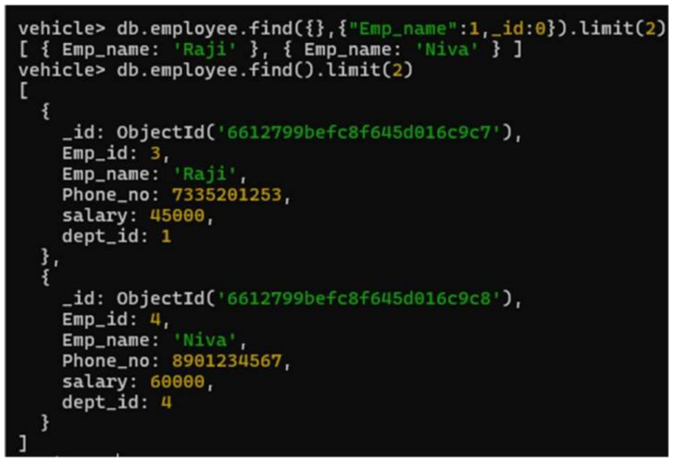
```
vehicle> db.employee.find({$or:[{"Emp_name":"Rithi"},{"Emp_id":4}]})
[
  {
    _id: ObjectId('6612799befc8f645d016c9c8'),
    Emp_id: 4,
    Emp_name: 'Niva',
    Phone_no: 8901234567,
    salary: 60000,
    dept_id: 4
  },
  {
    _id: ObjectId('66128339efc8f645d016c9cb'),
    Emp_id: 1,
    Emp_name: 'Rithi',
    Phone_no: 9345542103,
    salary: 50000
  }
]
```

- SKIP: To skip the top number of rows as specified in the query result
  db.collection_name.find().skip(number)



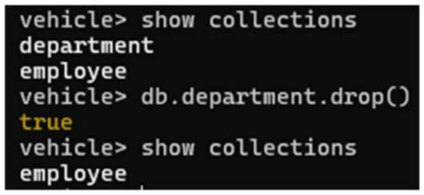
```
vehicle> db.employee.find().skip(1)
[
  {
    _id: ObjectId('6612799befc8f645d016c9c8'),
    Emp_id: 4,
    Emp_name: 'Niva',
    Phone_no: 8901234567,
    salary: 60000,
    dept_id: 4
  },
  {
    _id: ObjectId('661282fbefc8f645d016c9ca'),
    Emp_id: 2,
    Emp_name: 'Harini',
    Phone_no: 9003402381,
    salary: 45000
  },
  {
    _id: ObjectId('66128339efc8f645d016c9cb'),
    Emp_id: 1,
    Emp_name: 'Rithi',
    Phone_no: 9345542103,
    salary: 50000
  }
]
```

- Limit:
  To limit the records in MongoDB, you need to use limit() method. The method accepts one number type argument, which is the number of documents that you want to be displayed.
  Syntax
  The basic syntax of limit() method is as follows –
  ->db.COLLECTION_NAME.find().limit(NUMBER)



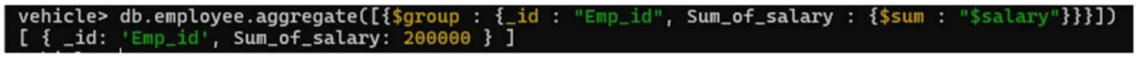
```
vehicle> db.employee.find({},{"Emp_name":1,_id:0}).limit(2)
[ { Emp_name: 'Raji' }, { Emp_name: 'Niva' } ]
vehicle> db.employee.find().limit(2)
[
  {
    _id: ObjectId('6612799befc8f645d016c9c7'),
    Emp_id: 3,
    Emp_name: 'Raji',
    Phone_no: 7335201253,
    salary: 45000,
    dept_id: 1
  },
  {
    _id: ObjectId('6612799befc8f645d016c9c8'),
    Emp_id: 4,
    Emp_name: 'Niva',
    Phone_no: 8901234567,
    salary: 60000,
    dept_id: 4
  }
]
```

- DROP COLLECTION: db.collection_name.drop()



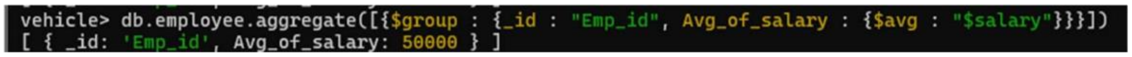
```
vehicle> show collections
department
employee
vehicle> db.department.drop()
true
vehicle> show collections
employee
```

- Aggregate functions:
  - ➢ $sum: Sums up the defined value from all documents in the collection.



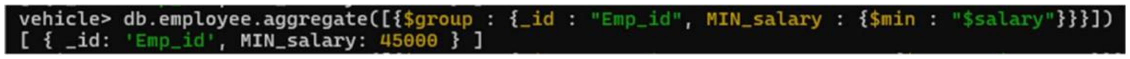
```
vehicle> db.employee.aggregate([{$group : {_id : "Emp_id", Sum_of_salary : {$sum : "$salary"}}}])
[ { _id: 'Emp_id', Sum_of_salary: 200000 } ]
```

  - ➢ $avg : Calculates the average of all given values from all documents in the collection.



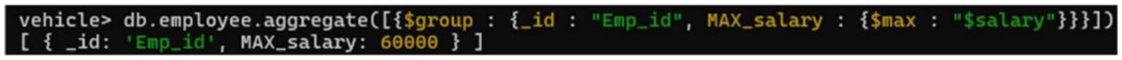
```
vehicle> db.employee.aggregate([{$group : {_id : "Emp_id", Avg_of_salary : {$avg : "$salary"}}}])
[ { _id: 'Emp_id', Avg_of_salary: 50000 } ]
```

  - ➢ $min: Gets the minimum of the corresponding values from all documents in the collection.



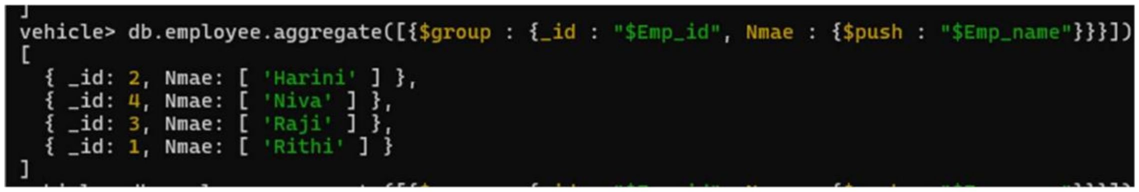
```
vehicle> db.employee.aggregate([{$group : {_id : "Emp_id", MIN_salary : {$min : "$salary"}}}])
[ { _id: 'Emp_id', MIN_salary: 45000 } ]
```

  - ➢ $max: Gets the maximum of the corresponding values from all documents in the collection.



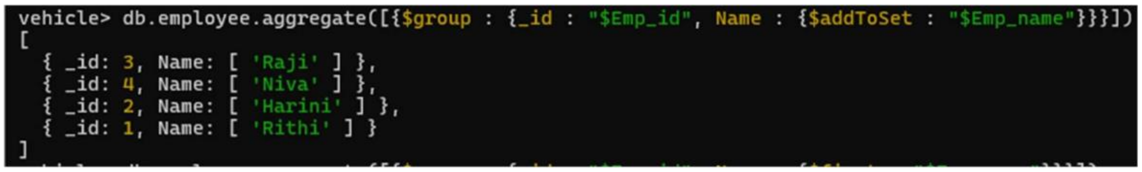
```
vehicle> db.employee.aggregate([{$group : {_id : "Emp_id", MAX_salary : {$max : "$salary"}}}])
[ { _id: 'Emp_id', MAX_salary: 60000 } ]
```

  - ➢ $push: Inserts the value to an array in the resulting document.



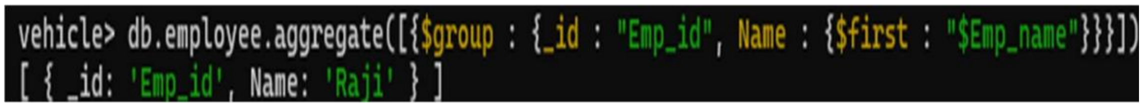
```
vehicle> db.employee.aggregate([{$group : {_id : "$Emp_id", Nmae : {$push : "$Emp_name"}}}])
[
  { _id: 2, Nmae: [ 'Harini' ] },
  { _id: 4, Nmae: [ 'Niva' ] },
  { _id: 3, Nmae: [ 'Raji' ] },
  { _id: 1, Nmae: [ 'Rithi' ] }
]
```

  - ➢ $addToSet: Inserts the value to an array in the resulting document but does not create duplicates.



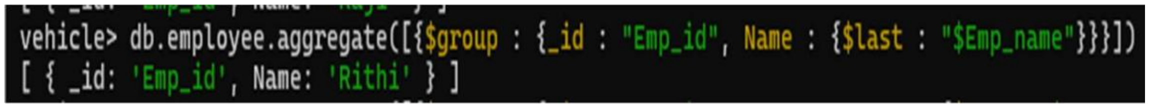
```
vehicle> db.employee.aggregate([{$group : {_id : "$Emp_id", Name : {$addToSet : "$Emp_name"}}}])
[
  { _id: 3, Name: [ 'Raji' ] },
  { _id: 4, Name: [ 'Niva' ] },
  { _id: 2, Name: [ 'Harini' ] },
  { _id: 1, Name: [ 'Rithi' ] }
]
```

- $first: Gets the first document from the source documents according to the grouping. Typically this makes only sense together with some previously applied "$sort"-stage.



```
vehicle> db.employee.aggregate([{$group : {_id : "Emp_id", Name : {$first : "$Emp_name"}}}])
[ { _id: 'Emp_id', Name: 'Raji' } ]
```

- $last: Gets the last document from the source documents according to the grouping. Typically this makes only sense together with some previously applied "$sort"-stage.



```
vehicle> db.employee.aggregate([{$group : {_id : "Emp_id", Name : {$last : "$Emp_name"}}}])
[ { _id: 'Emp_id', Name: 'Rithi' } ]
```

Result:

Thus, queries are executed and projection operations are performed successfully.