**Aggregation**

**Create Collection**

> db.createCollection("student1")

Db.student.drop();

**Collection name: student1**

> db.student1.find()

switched to db mydb

> db.student1.find();

{ "\_id" : ObjectId("55ee76a926f38b7cee2afc7c"), "name" : "aa", "Roll\_no" : 1, "s

ub" : "TOC", "Marks" : 80 }

{ "\_id" : ObjectId("55ee76be26f38b7cee2afc7d"), "name" : "bb", "Roll\_no" : 2, "s

ub" : "TOC", "Marks" : 100 }

{ "\_id" : ObjectId("55ee76d726f38b7cee2afc7e"), "name" : "cc", "Roll\_no" : 3, "s

ub" : "TOC", "Marks" : 90 }

{ "\_id" : ObjectId("55ee76ee26f38b7cee2afc7f"), "name" : "dd", "Roll\_no" : 4, "s

ub" : "TOC", "Marks" : 95 }

{ "\_id" : ObjectId("55ee771b26f38b7cee2afc80"), "name" : "aa", "Roll\_no" : 1, "s

ub" : "DBMS", "Marks" : 70 }

{ "\_id" : ObjectId("55ee773f26f38b7cee2afc81"), "name" : "ee", "Roll\_no" : 5, "s

ub" : "DBMS", "Marks" : 80 }

{ "\_id" : ObjectId("55ee776126f38b7cee2afc82"), "name" : "ff", "Roll\_no" : 6, "s

ub" : "DBMS", "Marks" : 10 }

**$project**

> db.student1.aggregate({$project:{name : 1}})

db.student1.aggregate({$project:{name : 1, age: 3}})

**$sort**

> db.student1.aggregate({$sort: {name: 1}})

> db.student1.aggregate({$sort: {name: -1}})

**$limit**

> db.student1.aggregate([{$project:{name : 1}},{$limit : 3}])

**$skip**

> db.student1.aggregate([{$project:{name : 1}},{$skip : 3}])

**Match and group**

**$sum**

> db.student1.aggregate( [ {$match:{sub:"TOC"}}, {$group:{\_id:"$name",TOTAL:{$sum:"$Marks"}}}])

> db.student.aggregate( [ {$match:{class:"TEA"}}, {$group:{\_id:"$name",TOTAL:{$sum:"$age"}}}])

db.college.aggregate( [ {$match:{name:"NBN"}}, {$group:{\_id:"$c\_id",TOTAL:{$sum:"$intake"}}}])

**$max**

db.student1.aggregate([{$match:{sub:"TOC"}}, {$group:{\_id:"$name",TOTAL:{$max:"$Marks"}}}])

**$min**

db.student1.aggregate([{$match:{sub:"TOC"}}, {$group:{\_id:"$Name",TOTAL:{$min:"$Marks"}}}])

**$push (top\_to\_bottom)**

db.student1.aggregate( [ {$match:{sub:"DBMS"}}, {$group:{\_id:"$name",TOTAL:{$push:"$Marks"}}} ])

**$addToSet (bottom\_to\_top)**

1. db.student1.aggregate( [ {$match:{sub:"DBMS"}}, {$group:{\_id:"$name",TOTAL:{$addToSet:"$Marks"}}}])
2. db.student1.aggregate( [ {$match:{Sub:"TOC"}}, {$group:{\_id:"$Roll\_no",TOTAL:{$addToSet:"$Marks"}}}])

**$first**

db.student1.aggregate( [ {$match:{sub:"TOC"}}, {$group:{\_id:"$name",TOTAL:{$first:"$Marks"}}}])

**$last**

db.student1.aggregate( [ {$match:{sub:"DBMS"}}, {$group:{\_id:"$name",TOTAL:{$last:"$Marks"}}}])

**distinct (different values in collections)**

1. > db.student1.distinct("sub")
2. >db.runCommand({distinct:"student1",key:"Roll\_no"});
3. > db.runCommand({distinct:"student1",key:"sub"})

**Count**

> db.student1.count()

| **Name** | **Description** |
| --- | --- |
| [$sum](https://docs.mongodb.com/manual/reference/operator/aggregation/sum/#grp._S_sum) | Returns a sum of numerical values. Ignores non-numeric values.  Changed in version 3.2: Available in both [$group](https://docs.mongodb.com/manual/reference/operator/aggregation/group/#pipe._S_group) and [$project](https://docs.mongodb.com/manual/reference/operator/aggregation/project/#pipe._S_project) stages. |
| [$avg](https://docs.mongodb.com/manual/reference/operator/aggregation/avg/#grp._S_avg) | Returns an average of numerical values. Ignores non-numeric values.  Changed in version 3.2: Available in both [$group](https://docs.mongodb.com/manual/reference/operator/aggregation/group/#pipe._S_group) and [$project](https://docs.mongodb.com/manual/reference/operator/aggregation/project/#pipe._S_project) stages. |
| [$first](https://docs.mongodb.com/manual/reference/operator/aggregation/first/#grp._S_first) | Returns a value from the first document for each group. Order is only defined if the documents are in a defined order.  Available in [$group](https://docs.mongodb.com/manual/reference/operator/aggregation/group/#pipe._S_group) stage only. |
| [$last](https://docs.mongodb.com/manual/reference/operator/aggregation/last/#grp._S_last) | Returns a value from the last document for each group. Order is only defined if the documents are in a defined order.  Available in [$group](https://docs.mongodb.com/manual/reference/operator/aggregation/group/#pipe._S_group) stage only. |
| [$max](https://docs.mongodb.com/manual/reference/operator/aggregation/max/#grp._S_max) | Returns the highest expression value for each group.  Changed in version 3.2: Available in both [$group](https://docs.mongodb.com/manual/reference/operator/aggregation/group/#pipe._S_group) and [$project](https://docs.mongodb.com/manual/reference/operator/aggregation/project/#pipe._S_project) stages. |
| [$min](https://docs.mongodb.com/manual/reference/operator/aggregation/min/#grp._S_min) | Returns the lowest expression value for each group.  Changed in version 3.2: Available in both [$group](https://docs.mongodb.com/manual/reference/operator/aggregation/group/#pipe._S_group) and [$project](https://docs.mongodb.com/manual/reference/operator/aggregation/project/#pipe._S_project) stages. |
| [$push](https://docs.mongodb.com/manual/reference/operator/aggregation/push/#grp._S_push) | Returns an array of expression values for each group.  Available in [$group](https://docs.mongodb.com/manual/reference/operator/aggregation/group/#pipe._S_group) stage only. |
| [$addToSet](https://docs.mongodb.com/manual/reference/operator/aggregation/addToSet/#grp._S_addToSet) | Returns an array of *unique* expression values for each group. Order of the array elements is undefined.  Available in [$group](https://docs.mongodb.com/manual/reference/operator/aggregation/group/#pipe._S_group) stage only. |