**##3**

1.db.restaurants.find({ "borough": "Queens" , $and : [{ "cuisine" : "Chinese" }] }).count();

o/p:

728

2.db.restaurants.aggregate([   { $unwind: '$grades' },   { $group: { \_id:null, max: { $max: '$grades.score' } } },   { $project: { max: 1, \_id:'$\_id' } } ])

3.db.restaurants.find({"borough": "Manhattan", $and : [db.restaurants.insertOne({ grade: "A", score: 7, date: ISODate()})]});

5. db.restaurants.find({ "grades.date": {$gte: ISODate("2014-03-11T00:00:00Z"), $lte: ISODate("2014-02-11T00:00:00Z")}, "grades.grade":"B", "cuisine": "Seafood"}, {"\_id" : 1,"name":1});

**##4**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

**1. Create an index which will be used by this query and provide proof (from explain() or Compass UI) that the index is indeed used by the winning plan:**

**db.restaurants.find({ name: "Glorious Food" });**

Before Index:

db.restaurants.find({ name: "Glorious Food" }).explain("executionStats")

{

"queryPlanner" : {

"plannerVersion" : 1,

"namespace" : "frontcamp.restaurants",

"indexFilterSet" : false,

"parsedQuery" : {

"name" : {

"$eq" : "Glorious Food"

}

},

"winningPlan" : {

"stage" : "COLLSCAN",

"filter" : {

"name" : {

"$eq" : "Glorious Food"

}

},

"direction" : "forward"

},

"rejectedPlans" : [ ]

},

"executionStats" : {

"executionSuccess" : true,

"nReturned" : 1,

"executionTimeMillis" : 10,

"totalKeysExamined" : 0,

"totalDocsExamined" : 25361,

"executionStages" : {

"stage" : "COLLSCAN",

"filter" : {

"name" : {

"$eq" : "Glorious Food"

}

},

"nReturned" : 1,

"executionTimeMillisEstimate" : 10,

"works" : 25363,

"advanced" : 1,

"needTime" : 25361,

"needYield" : 0,

"saveState" : 198,

"restoreState" : 198,

"isEOF" : 1,

"invalidates" : 0,

"direction" : "forward",

"docsExamined" : 25361

}

},

"serverInfo" : {

"host" : "EPINHYDW0756",

"port" : 27017,

"version" : "4.0.9",

"gitVersion" : "fc525e2d9b0e4bceff5c2201457e564362909765"

},

"ok" : 1

}

> db.restaurants.createIndex( { name: 1 } )

{

"createdCollectionAutomatically" : false,

"numIndexesBefore" : 1,

"numIndexesAfter" : 2,

"ok" : 1

}

After Index

db.restaurants.createIndex( { name: 1 } )

{

"createdCollectionAutomatically" : false,

"numIndexesBefore" : 1,

"numIndexesAfter" : 2,

"ok" : 1

}

db.restaurants.find({ name: "Glorious Food" }).explain("executionStats")

{

"queryPlanner" : {

"plannerVersion" : 1,

"namespace" : "frontcamp.restaurants",

"indexFilterSet" : false,

"parsedQuery" : {

"name" : {

"$eq" : "Glorious Food"

}

},

"winningPlan" : {

"stage" : "FETCH",

"inputStage" : {

"stage" : "IXSCAN",

"keyPattern" : {

"name" : 1

},

"indexName" : "name\_1",

"isMultiKey" : false,

"multiKeyPaths" : {

"name" : [ ]

},

"isUnique" : false,

"isSparse" : false,

"isPartial" : false,

"indexVersion" : 2,

"direction" : "forward",

"indexBounds" : {

"name" : [

"[\"Glorious Food\", \"Glorious Food\"]"

]

}

}

},

"rejectedPlans" : [ ]

},

"executionStats" : {

"executionSuccess" : true,

"nReturned" : 1,

"executionTimeMillis" : 2,

"totalKeysExamined" : 1,

"totalDocsExamined" : 1,

"executionStages" : {

"stage" : "FETCH",

"nReturned" : 1,

"executionTimeMillisEstimate" : 0,

"works" : 2,

"advanced" : 1,

"needTime" : 0,

"needYield" : 0,

"saveState" : 0,

"restoreState" : 0,

"isEOF" : 1,

"invalidates" : 0,

"docsExamined" : 1,

"alreadyHasObj" : 0,

"inputStage" : {

"stage" : "IXSCAN",

"nReturned" : 1,

"executionTimeMillisEstimate" : 0,

"works" : 2,

"advanced" : 1,

"needTime" : 0,

"needYield" : 0,

"saveState" : 0,

"restoreState" : 0,

"isEOF" : 1,

"invalidates" : 0,

"keyPattern" : {

"name" : 1

},

"indexName" : "name\_1",

"isMultiKey" : false,

"multiKeyPaths" : {

"name" : [ ]

},

"isUnique" : false,

"isSparse" : false,

"isPartial" : false,

"indexVersion" : 2,

"direction" : "forward",

"indexBounds" : {

"name" : [

"[\"Glorious Food\", \"Glorious Food\"]"

]

},

"keysExamined" : 1,

"seeks" : 1,

"dupsTested" : 0,

"dupsDropped" : 0,

"seenInvalidated" : 0

}

}

},

"serverInfo" : {

"host" : "EPINHYDW0756",

"port" : 27017,

"version" : "4.0.9",

"gitVersion" : "fc525e2d9b0e4bceff5c2201457e564362909765"

},

"ok" : 1

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

**2. Drop index from task 4.1**

> db.restaurants.dropIndex( { name: 1 } )

{ "nIndexesWas" : 2, "ok" : 1 }

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

**3. Create an index to make this query covered and provide proof (from explain() or Compass UI) that it is indeed covered:**

**db.restaurants.find({ restaurant\_id: "41098650" }, { \_id: 0, borough: 1 })**

**Before:**

> db.restaurants.find({ restaurant\_id: "41098650" }, { \_id: 0, borough: 1 }).explain("executionStats")

{

"queryPlanner" : {

"plannerVersion" : 1,

"namespace" : "frontcamp.restaurants",

"indexFilterSet" : false,

"parsedQuery" : {

"restaurant\_id" : {

"$eq" : "41098650"

}

},

"winningPlan" : {

"stage" : "PROJECTION",

"transformBy" : {

"\_id" : 0,

"borough" : 1

},

"inputStage" : {

"stage" : "COLLSCAN",

"filter" : {

"restaurant\_id" : {

"$eq" : "41098650"

}

},

"direction" : "forward"

}

},

"rejectedPlans" : [ ]

},

"executionStats" : {

"executionSuccess" : true,

"nReturned" : 1,

"executionTimeMillis" : 11,

"totalKeysExamined" : 0,

"totalDocsExamined" : 25361,

"executionStages" : {

"stage" : "PROJECTION",

"nReturned" : 1,

"executionTimeMillisEstimate" : 10,

"works" : 25363,

"advanced" : 1,

"needTime" : 25361,

"needYield" : 0,

"saveState" : 198,

"restoreState" : 198,

"isEOF" : 1,

"invalidates" : 0,

"transformBy" : {

"\_id" : 0,

"borough" : 1

},

"inputStage" : {

"stage" : "COLLSCAN",

"filter" : {

"restaurant\_id" : {

"$eq" : "41098650"

}

},

"nReturned" : 1,

"executionTimeMillisEstimate" : 10,

"works" : 25363,

"advanced" : 1,

"needTime" : 25361,

"needYield" : 0,

"saveState" : 198,

"restoreState" : 198,

"isEOF" : 1,

"invalidates" : 0,

"direction" : "forward",

"docsExamined" : 25361

}

}

},

"serverInfo" : {

"host" : "EPINHYDW0756",

"port" : 27017,

"version" : "4.0.9",

"gitVersion" : "fc525e2d9b0e4bceff5c2201457e564362909765"

},

"ok" : 1

}

**After:**

**> db.restaurants.createIndex( { restaurant\_id:1 } )**

**{**

**"createdCollectionAutomatically" : false,**

**"numIndexesBefore" : 1,**

**"numIndexesAfter" : 2,**

**"ok" : 1**

**}**

**> db.restaurants.find({ restaurant\_id: "41098650" }, { \_id: 0, borough: 1 }).explain("executionStats")**

**{**

**"queryPlanner" : {**

**"plannerVersion" : 1,**

**"namespace" : "frontcamp.restaurants",**

**"indexFilterSet" : false,**

**"parsedQuery" : {**

**"restaurant\_id" : {**

**"$eq" : "41098650"**

**}**

**},**

**"winningPlan" : {**

**"stage" : "PROJECTION",**

**"transformBy" : {**

**"\_id" : 0,**

**"borough" : 1**

**},**

**"inputStage" : {**

**"stage" : "FETCH",**

**"inputStage" : {**

**"stage" : "IXSCAN",**

**"keyPattern" : {**

**"restaurant\_id" : 1**

**},**

**"indexName" : "restaurant\_id\_1",**

**"isMultiKey" : false,**

**"multiKeyPaths" : {**

**"restaurant\_id" : [ ]**

**},**

**"isUnique" : false,**

**"isSparse" : false,**

**"isPartial" : false,**

**"indexVersion" : 2,**

**"direction" : "forward",**

**"indexBounds" : {**

**"restaurant\_id" : [**

**"[\"41098650\", \"41098650\"]"**

**]**

**}**

**}**

**}**

**},**

**"rejectedPlans" : [ ]**

**},**

**"executionStats" : {**

**"executionSuccess" : true,**

**"nReturned" : 1,**

**"executionTimeMillis" : 0,**

**"totalKeysExamined" : 1,**

**"totalDocsExamined" : 1,**

**"executionStages" : {**

**"stage" : "PROJECTION",**

**"nReturned" : 1,**

**"executionTimeMillisEstimate" : 0,**

**"works" : 2,**

**"advanced" : 1,**

**"needTime" : 0,**

**"needYield" : 0,**

**"saveState" : 0,**

**"restoreState" : 0,**

**"isEOF" : 1,**

**"invalidates" : 0,**

**"transformBy" : {**

**"\_id" : 0,**

**"borough" : 1**

**},**

**"inputStage" : {**

**"stage" : "FETCH",**

**"nReturned" : 1,**

**"executionTimeMillisEstimate" : 0,**

**"works" : 2,**

**"advanced" : 1,**

**"needTime" : 0,**

**"needYield" : 0,**

**"saveState" : 0,**

**"restoreState" : 0,**

**"isEOF" : 1,**

**"invalidates" : 0,**

**"docsExamined" : 1,**

**"alreadyHasObj" : 0,**

**"inputStage" : {**

**"stage" : "IXSCAN",**

**"nReturned" : 1,**

**"executionTimeMillisEstimate" : 0,**

**"works" : 2,**

**"advanced" : 1,**

**"needTime" : 0,**

**"needYield" : 0,**

**"saveState" : 0,**

**"restoreState" : 0,**

**"isEOF" : 1,**

**"invalidates" : 0,**

**"keyPattern" : {**

**"restaurant\_id" : 1**

**},**

**"indexName" : "restaurant\_id\_1",**

**"isMultiKey" : false,**

**"multiKeyPaths" : {**

**"restaurant\_id" : [ ]**

**},**

**"isUnique" : false,**

**"isSparse" : false,**

**"isPartial" : false,**

**"indexVersion" : 2,**

**"direction" : "forward",**

**"indexBounds" : {**

**"restaurant\_id" : [**

**"[\"41098650\", \"41098650\"]"**

**]**

**},**

**"keysExamined" : 1,**

**"seeks" : 1,**

**"dupsTested" : 0,**

**"dupsDropped" : 0,**

**"seenInvalidated" : 0**

**}**

**}**

**}**

**},**

**"serverInfo" : {**

**"host" : "EPINHYDW0756",**

**"port" : 27017,**

**"version" : "4.0.9",**

**"gitVersion" : "fc525e2d9b0e4bceff5c2201457e564362909765"**

**},**

**"ok" : 1**

**}**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

**4**. **Create a partial index on cuisine field which will be used only when filtering on borough equal to “Staten Island”:**

db.restaurants.createIndex(

{ cuisine: 1 },

{ partialFilterExpression: { borough: "Staten Island" } }

)

----

{

"createdCollectionAutomatically" : false,

"numIndexesBefore" : 2,

"numIndexesAfter" : 3,

"ok" : 1

}

5. Create an index to make query from task 3.4 covered and provide proof (from explain() or Compass UI) that it is indeed covered

db.restaurants.find({ borough: "Staten Island", cuisine: "American" }).explain("executionStats")