

3. Querying Restaurants Collection:

1. How many “Chinese” (cuisine) restaurants are in “Queens” (borough)?

Query:

```
db.restaurants.find({"borough": "Queens", "cuisine": "Chinese"}).count()
```

Result: 728

2. What is the `_id` of the restaurant which has the grade with the highest ever score?

Query: `db.restaurants.find({}, {restaurant_id:1}).sort({"grades.score":-1}).limit(1).pretty()`

Result:

```
{
  "_id" : ObjectId("5de538cd2c61c0d4fa8c190c"),
  "restaurant_id" : "40372466"
}
```

```
@(Shell):1:33
> db.restaurants.find({}, {restaurant_id:1}).sort({"grades.score":-1}).limit(1).pretty()
{
  "_id" : ObjectId("5de538cd2c61c0d4fa8c190c"),
  "restaurant_id" : "40372466"
}
>
```

3. Add a grade { grade: "A", score: 7, date: ISODate() } to every restaurant in “Manhattan” (borough).

Query:

```
db.restaurants.updateMany({ borough: 'Manhattan' }, { $push: { grades: { grade: "A", score: 7, date: ISODate() } } })
```

Result:

```
{ "acknowledged" : true, "matchedCount" : 10259, "modifiedCount" : 10259 }
```

```
> db.restaurants.updateMany({ borough: 'Manhattan' }, { $push: { grades: { grade: "A", score: 7, date: ISODate() } } })
{ "acknowledged" : true, "matchedCount" : 10259, "modifiedCount" : 10259 }
>
```

4. What are the names of the restaurants which have a grade at index 8 with score less than 7? Use projection to include only names without `_id`.

Query:

```
db.restaurants.find({ 'grades.8.score': { $lt: 7 }, { _id: 0, name: 1 }).pretty()
```

Result:

```
{ "name" : "Silver Krust West Indian Restaurant" }
{ "name" : "Pure Food" }
```

```
> db.restaurants.find({ 'grades.8.score': { $lt: 7 } }, { _id: 0, name: 1 }).pretty()
{ "name" : "Silver Krust West Indian Restaurant" }
{ "name" : "Pure Food" }
>
```

5. What are `_id` and borough of “Seafood” (cuisine) restaurants which received at least one “B” grade in period from 2014-02-01 to 2014-03-01? Use projection to include only `_id` and borough.

Query:

```
db.restaurants.find({cuisine: 'Seafood', grades:{$elemMatch: { date: { $gte: ISODate("2014-02-01"),
$lt: ISODate("2014-03-01") }, grade: 'B' } }}, { borough: 1 }).pretty()
```

Result:

```
{ "_id" : ObjectId("5de538ce2c61c0d4fa8c4d09"), "borough" : "Bronx" }
{ "_id" : ObjectId("5de538ce2c61c0d4fa8c4f82"), "borough" : "Manhattan" }
```

```
> db.restaurants.find({cuisine: 'Seafood', grades:{$elemMatch: { date: { $gte: ISODate("2014-02-01"), $lt: ISODate("2014-03-01") }, grade: 'B' } }}, { borough: 1 }).pretty()
{ "_id" : ObjectId("5de538ce2c61c0d4fa8c4d09"), "borough" : "Bronx" }
{ "_id" : ObjectId("5de538ce2c61c0d4fa8c4f82"), "borough" : "Manhattan" }
>
```

4. Indexing Restaurants:

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" })
```

Query:

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

Result:

```
{
  "createdCollectionAutomatically" : false,
  "numIndexesBefore" : 1,
  "numIndexesAfter" : 2,
  "ok" : 1
}
```

```
> db.restaurants.createIndex({ name: 1 })
{
  "createdCollectionAutomatically" : false,
  "numIndexesBefore" : 1,
  "numIndexesAfter" : 2,
  "ok" : 1
}
```

Query:

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

Result:

```
{
  "queryPlanner" : {
    "plannerVersion" : 1,
    "namespace" : "frontcamp.restaurants",
    "indexFilterSet" : false,
    "parsedQuery" : {
      "name" : {
        "$eq" : "Glorious Food"
      }
    },
    "queryHash" : "01AEE5EC",
    "planCacheKey" : "4C5AEA2C",
    "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" : [ ]
},
"serverInfo" : {
  "host" : "EPBYMINW8582",
  "port" : 27017,
  "version" : "4.2.1",
  "gitVersion" : "edf6d45851c0b9ee15548f0f847df141764a317e"
},
"ok" : 1
```

```

> db.restaurants.find({ name: "Glorious Food" }).explain()
{
  "queryPlanner" : {
    "plannerVersion" : 1,
    "namespace" : "frontcamp.restaurants",
    "indexFilterSet" : false,
    "parsedQuery" : {
      "name" : {
        "$eq" : "Glorious Food"
      }
    },
    "queryHash" : "01AEE5EC",
    "planCacheKey" : "4C5AEA2C",
    "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" : [ ]
    },
    "serverInfo" : {
      "host" : "EPBYMINW8582",
      "port" : 27017,
      "version" : "4.2.1",
      "gitVersion" : "edf6d45851c0b9ee15548f0f847df141764a317e"
    },
    "ok" : 1
  }
}

```

2. Drop index from task 4.1

Query:

```
db.restaurants.dropIndex('name_1')
```

Result:

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

```
}  
> db.restaurants.dropIndex('name_1')  
{ "nIndexesWas" : 2, "ok" : 1 }  
>
```

Query:

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

Result:

```
{  
  "queryPlanner" : {  
    "plannerVersion" : 1,  
    "namespace" : "frontcamp.restaurants",  
    "indexFilterSet" : false,  
    "parsedQuery" : {  
      "name" : {  
        "$eq" : "Glorious Food"  
      }  
    },  
    "queryHash" : "01AEE5EC",  
    "planCacheKey" : "01AEE5EC",  
    "winningPlan" : {  
      "stage" : "COLLSCAN",  
      "filter" : {  
        "name" : {  
          "$eq" : "Glorious Food"  
        }  
      },  
      "direction" : "forward"  
    },  
    "rejectedPlans" : []  
  },  
  "serverInfo" : {
```

```

    "host" : "EPBYMINW8582",
    "port" : 27017,
    "version" : "4.2.1",
    "gitVersion" : "edf6d45851c0b9ee15548f0f847df141764a317e"
  },
  "ok" : 1
}

```

```

> db.restaurants.find({ name: "Glorious Food" }).explain()
{
  "queryPlanner" : {
    "plannerVersion" : 1,
    "namespace" : "frontcamp.restaurants",
    "indexFilterSet" : false,
    "parsedQuery" : {
      "name" : {
        "$eq" : "Glorious Food"
      }
    },
    "queryHash" : "01AEE5EC",
    "planCacheKey" : "01AEE5EC",
    "winningPlan" : {
      "stage" : "COLLSCAN",
      "filter" : {
        "name" : {
          "$eq" : "Glorious Food"
        }
      }
    },
    "direction" : "forward"
  },
  "rejectedPlans" : [ ]
},
"serverInfo" : {
  "host" : "EPBYMINW8582",
  "port" : 27017,
  "version" : "4.2.1",
  "gitVersion" : "edf6d45851c0b9ee15548f0f847df141764a317e"
},
"ok" : 1
}

```

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

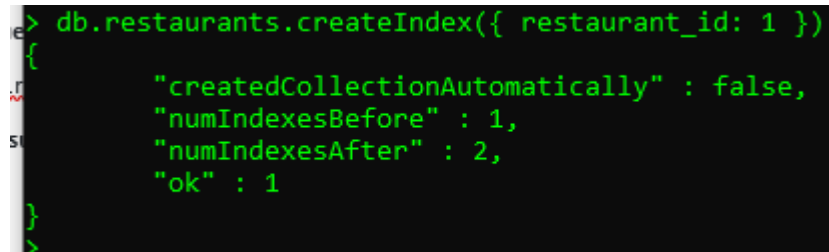
```
db.restaurants.find({ restaurant_id: "41098650" }, { _id: 0, borough: 1 })
```

Query:

```
db.restaurants.createIndex({ restaurant_id: 1 })
```

Result:

```
{
  "createdCollectionAutomatically" : false,
  "numIndexesBefore" : 1,
  "numIndexesAfter" : 2,
  "ok" : 1
}
```



```
> db.restaurants.createIndex({ restaurant_id: 1 })
{
  "createdCollectionAutomatically" : false,
  "numIndexesBefore" : 1,
  "numIndexesAfter" : 2,
  "ok" : 1
}
```

Query:

```
db.restaurants.find({ restaurant_id: "41098650" }, { _id: 0, borough: 1 }).explain()
```

Result:

```
{
  "queryPlanner" : {
    "plannerVersion" : 1,
    "namespace" : "frontcamp.restaurants",
    "indexFilterSet" : false,
    "parsedQuery" : {
      "restaurant_id" : {
        "$eq" : "41098650"
      }
    },
    "queryHash" : "11B8AFCC",
    "planCacheKey" : "A2837C36",
    "winningPlan" : {
      "stage" : "PROJECTION_SIMPLE",
      "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" : [ ]
  },
  "serverInfo" : {
```

```
"host" : "EPBYMINW8582",  
"port" : 27017,  
"version" : "4.2.1",  
"gitVersion" : "edf6d45851c0b9ee15548f0f847df141764a317e"  
},  
"ok" : 1  
}
```

```

> db.restaurants.find({ restaurant_id: "41098650" }, { _id: 0, borough: 1 }).explain()
{
  "queryPlanner" : {
    "plannerVersion" : 1,
    "namespace" : "frontcamp.restaurants",
    "indexFilterSet" : false,
    "parsedQuery" : {
      "restaurant_id" : {
        "$eq" : "41098650"
      }
    },
    "queryHash" : "11B8AFCC",
    "planCacheKey" : "A2837C36",
    "winningPlan" : {
      "stage" : "PROJECTION_SIMPLE",
      "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" : [ ]
  },
  "serverInfo" : {
    "host" : "EPBYMINW8582",
    "port" : 27017,
    "version" : "4.2.1",
    "gitVersion" : "edf6d45851c0b9ee15548f0f847df141764a317e"
  },
  "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.find({ borough: "Staten Island", cuisine: "American" })` – uses index

`db.restaurants.find({ borough: "Staten Island", name: "Bagel Land" })` – does not use index

db.restaurants.find({ borough: "Queens", cuisine: "Pizza" }) – does not use index

Query:

db.restaurants.createIndex({ cuisine: 1 }, { partialFilterExpression: { borough: 'Staten Island'} })

Result:

```
{
  "createdCollectionAutomatically" : false,
  "numIndexesBefore" : 2,
  "numIndexesAfter" : 3,
  "ok" : 1
}
```

```
> db.restaurants.createIndex({ cuisine: 1 }, { partialFilterExpression: { borough: 'Staten Island'} })
{
  "createdCollectionAutomatically" : false,
  "numIndexesBefore" : 2,
  "numIndexesAfter" : 3,
  "ok" : 1
}
```

Query:

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

Result:

```
{
  "queryPlanner" : {
    "plannerVersion" : 1,
    "namespace" : "frontcamp.restaurants",
    "indexFilterSet" : false,
    "parsedQuery" : {
      "$and" : [
        {
          "borough" : {
            "$eq" : "Staten Island"
          }
        },
        {
          "cuisine" : {
```

```
        "$seq" : "American"
      }
    }
  ]
},
"queryHash" : "DBDC0200",
"planCacheKey" : "C53EF8BB",
"winningPlan" : {
  "stage" : "FETCH",
  "filter" : {
    "borough" : {
      "$seq" : "Staten Island"
    }
  },
  "inputStage" : {
    "stage" : "IXSCAN",
    "keyPattern" : {
      "cuisine" : 1
    },
    "indexName" : "cuisine_1",
    "isMultiKey" : false,
    "multiKeyPaths" : {
      "cuisine" : [ ]
    },
    "isUnique" : false,
    "isSparse" : false,
    "isPartial" : true,
    "indexVersion" : 2,
    "direction" : "forward",
    "indexBounds" : {
```

```
        "cuisine" : [
            ["American\\", "American\\"]
        ]
    }
}

},
"rejectedPlans" : [ ]
},
"serverInfo" : {
    "host" : "EPBYMINW8582",
    "port" : 27017,
    "version" : "4.2.1",
    "gitVersion" : "edf6d45851c0b9ee15548f0f847df141764a317e"
},
"ok" : 1
}
```

```

{
  "queryPlanner" : {
    "plannerVersion" : 1,
    "namespace" : "frontcamp.restaurants",
    "indexFilterSet" : false,
    "parsedQuery" : {
      "$and" : [
        {
          "borough" : {
            "$eq" : "Staten Island"
          }
        },
        {
          "cuisine" : {
            "$eq" : "American"
          }
        }
      ]
    },
    "queryHash" : "DBDC0200",
    "planCacheKey" : "C53EF8BB",
    "winningPlan" : {
      "stage" : "FETCH",
      "filter" : {
        "borough" : {
          "$eq" : "Staten Island"
        }
      },
      "inputStage" : {
        "stage" : "IXSCAN",
        "keyPattern" : {
          "cuisine" : 1
        },
        "indexName" : "cuisine_1",
        "isMultiKey" : false,
        "multiKeyPaths" : {
          "cuisine" : [ ]
        },
        "isUnique" : false,
        "isSparse" : false,
        "isPartial" : true,
        "indexVersion" : 2,
        "direction" : "forward",
        "indexBounds" : {
          "cuisine" : [
            ["American\\", "American\\"]
          ]
        }
      }
    },
    "rejectedPlans" : [ ]
  },
  "serverInfo" : {
    "host" : "EPBYMINW8582",
    "port" : 27017,
    "version" : "4.2.1",
    "gitVersion" : "edf6d45851c0b9ee15548f0f847df141764a317e"
  },
  "ok" : 1
}

```

Query:

```
db.restaurants.find({ borough: "Staten Island", name: "Bagel Land" }).explain()
```

Result:

```
{
  "queryPlanner" : {
    "plannerVersion" : 1,
    "namespace" : "frontcamp.restaurants",
    "indexFilterSet" : false,
    "parsedQuery" : {
      "$and" : [
        {
          "borough" : {
            "$eq" : "Staten Island"
          }
        },
        {
          "name" : {
            "$eq" : "Bagel Land"
          }
        }
      ]
    },
    "queryHash" : "D9E6DF40",
    "planCacheKey" : "7175E33A",
    "winningPlan" : {
      "stage" : "COLLSCAN",
      "filter" : {
        "$and" : [
          {
```



```
        "borough" : {
            "$eq" : "Staten Island"
        }
    },
    {
        "name" : {
            "$eq" : "Bagel Land"
        }
    }
]
},
"direction" : "forward"
},
"rejectedPlans" : [ ]
},
"serverInfo" : {
    "host" : "EPBYMINW8582",
    "port" : 27017,
    "version" : "4.2.1",
    "gitVersion" : "edf6d45851c0b9ee15548f0f847df141764a317e"
},
"ok" : 1
}
```

```

> db.restaurants.find({ borough: "Staten Island", name: "Bagel Land" }).explain()
{
  "queryPlanner" : {
    "plannerVersion" : 1,
    "namespace" : "frontcamp.restaurants",
    "indexFilterSet" : false,
    "parsedQuery" : {
      "$and" : [
        {
          "borough" : {
            "$eq" : "Staten Island"
          }
        },
        {
          "name" : {
            "$eq" : "Bagel Land"
          }
        }
      ]
    },
    "queryHash" : "D9E6DF40",
    "planCacheKey" : "7175E33A",
    "winningPlan" : {
      "stage" : "COLLSCAN",
      "filter" : {
        "$and" : [
          {
            "borough" : {
              "$eq" : "Staten Island"
            }
          },
          {
            "name" : {
              "$eq" : "Bagel Land"
            }
          }
        ]
      }
    },
    "direction" : "forward"
  },
  "rejectedPlans" : [ ]
},
  "serverInfo" : {
    "host" : "EPBYMINW8582",
    "port" : 27017,
    "version" : "4.2.1",
    "gitVersion" : "edf6d45851c0b9ee15548f0f847df141764a317e"
  },
  "ok" : 1
}
>

```

Query:

```
db.restaurants.find({ borough: "Queens", cuisine: "Pizza" }).explain()
```

Result:

```

{
  "queryPlanner" : {
    "plannerVersion" : 1,

```

```
"namespace" : "frontcamp.restaurants",
"indexFilterSet" : false,
"parsedQuery" : {
  "$and" : [
    {
      "borough" : {
        "$eq" : "Queens"
      }
    },
    {
      "cuisine" : {
        "$eq" : "Pizza"
      }
    }
  ]
},
"queryHash" : "DBDC0200",
"planCacheKey" : "037B0B97",
"winningPlan" : {
  "stage" : "COLLSCAN",
  "filter" : {
    "$and" : [
      {
        "borough" : {
          "$eq" : "Queens"
        }
      },
      {
        "cuisine" : {
          "$eq" : "Pizza"
        }
      }
    ]
  }
}
```

```
        }
    }
]
},
"direction" : "forward"
},
"rejectedPlans" : [ ]
},
"serverInfo" : {
    "host" : "EPBYMINW8582",
    "port" : 27017,
    "version" : "4.2.1",
    "gitVersion" : "edf6d45851c0b9ee15548f0f847df141764a317e"
},
"ok" : 1
}
```

```

> db.restaurants.find({ borough: "Queens", cuisine: "Pizza" }).explain()
{
  "queryPlanner" : {
    "plannerVersion" : 1,
    "namespace" : "frontcamp.restaurants",
    "indexFilterSet" : false,
    "parsedQuery" : {
      "$and" : [
        {
          "borough" : {
            "$eq" : "Queens"
          }
        },
        {
          "cuisine" : {
            "$eq" : "Pizza"
          }
        }
      ]
    },
    "queryHash" : "DBDC0200",
    "planCacheKey" : "037B0B97",
    "winningPlan" : {
      "stage" : "COLLSCAN",
      "filter" : {
        "$and" : [
          {
            "borough" : {
              "$eq" : "Queens"
            }
          },
          {
            "cuisine" : {
              "$eq" : "Pizza"
            }
          }
        ]
      }
    },
    "direction" : "forward"
  },
  "rejectedPlans" : [ ]
},
"serverInfo" : {
  "host" : "EPBYMINW8582",
  "port" : 27017,
  "version" : "4.2.1",
  "gitVersion" : "edf6d45851c0b9ee15548f0f847df141764a317e"
},
"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

Query:

```
db.restaurants.createIndex({ 'grades.8.score': 1 })
```

Result:

```
{
  "createdCollectionAutomatically" : false,
  "numIndexesBefore" : 3,
  "numIndexesAfter" : 4,
  "ok" : 1
}
```

```
> db.restaurants.createIndex({ 'grades.8.score': 1 })
{
  "createdCollectionAutomatically" : false,
  "numIndexesBefore" : 3,
  "numIndexesAfter" : 4,
  "ok" : 1
}
```

Query:

```
db.restaurants.find({ 'grades.8.score': { $lt: 7 } }, { _id: 0, name: 1 }).explain()
```

Result:

```
{
  "queryPlanner" : {
    "plannerVersion" : 1,
    "namespace" : "frontcamp.restaurants",
    "indexFilterSet" : false,
    "parsedQuery" : {
      "grades.8.score" : {
        "$lt" : 7
      }
    },
    "queryHash" : "03034A2A",
    "planCacheKey" : "3B1B8634",
    "winningPlan" : {
      "stage" : "PROJECTION_SIMPLE",
```

```
"transformBy" : {
  "_id" : 0,
  "name" : 1
},
"inputStage" : {
  "stage" : "FETCH",
  "inputStage" : {
    "stage" : "IXSCAN",
    "keyPattern" : {
      "grades.8.score" : 1
    },
    "indexName" : "grades.8.score_1",
    "isMultiKey" : true,
    "multiKeyPaths" : {
      "grades.8.score" : [
        "grades"
      ]
    },
    "isUnique" : false,
    "isSparse" : false,
    "isPartial" : false,
    "indexVersion" : 2,
    "direction" : "forward",
    "indexBounds" : {
      "grades.8.score" : [
        "[-inf.0, 7.0)"
      ]
    }
  }
}
```

```
    },
    "rejectedPlans" : [ ]
  },
  "serverInfo" : {
    "host" : "EPBYMINW8582",
    "port" : 27017,
    "version" : "4.2.1",
    "gitVersion" : "edf6d45851c0b9ee15548f0f847df141764a317e"
  },
  "ok" : 1
}
```



```

> db.restaurants.find({ 'grades.8.score': { $lt: 7 }}, { _id: 0, name: 1 }).explain()
{
  "queryPlanner" : {
    "plannerVersion" : 1,
    "namespace" : "frontcamp.restaurants",
    "indexFilterSet" : false,
    "parsedQuery" : {
      "grades.8.score" : {
        "$lt" : 7
      }
    },
    "queryHash" : "03034A2A",
    "planCacheKey" : "3B1B8634",
    "winningPlan" : {
      "stage" : "PROJECTION_SIMPLE",
      "transformBy" : {
        "_id" : 0,
        "name" : 1
      },
      "inputStage" : {
        "stage" : "FETCH",
        "inputStage" : {
          "stage" : "IXSCAN",
          "keyPattern" : {
            "grades.8.score" : 1
          },
          "indexName" : "grades.8.score_1",
          "isMultiKey" : true,
          "multiKeyPaths" : {
            "grades.8.score" : [
              "grades"
            ]
          },
          "isUnique" : false,
          "isSparse" : false,
          "isPartial" : false,
          "indexVersion" : 2,
          "direction" : "forward",
          "indexBounds" : {
            "grades.8.score" : [
              "[-inf.0, 7.0)"
            ]
          }
        }
      }
    },
    "rejectedPlans" : [ ]
  },
  "serverInfo" : {
    "host" : "EPBYMINW8582",
    "port" : 27017,
    "version" : "4.2.1",
    "gitVersion" : "edf6d45851c0b9ee15548f0f847df141764a317e"
  },
  "ok" : 1
}

```

