1. Använd en databas med namn myDB.

|  |
| --- |
| use myDB |
| switched to db myDB |

1. Skapa ett dokument med innehåll FirstName: ”Selma”, LastName: Lagerlöf och sätt in det i en kollektion med namn ”authors”.

|  |
| --- |
| db.authors.insertOne(  { "FirstName": "Selma", "LastName": "Lagerlöf" }  ) |
| {  "acknowledged" : true,  "insertedId" : ObjectId("60acb3aa07db3fe8c8d672fa")  } |

1. Lägg till ytterligare ett dokument i ”authors” med FirstName: ”August”, LastName: ”Bondeson”, Birth: 1854

|  |
| --- |
| db.authors.insertOne({  "FirstName": "August",  "LastName": "Bondeson",  "Birth": 1854  }) |
| {  "acknowledged" : true,  "insertedId" : ObjectId("60acb4f907db3fe8c8d672fb")  } |

1. Uppdatera dokumentet för August Bondeson och lägg till Death: 1906

|  |
| --- |
| db.authors.updateOne(  { "\_id": ObjectId("60acb4f907db3fe8c8d672fb") },  { "$set": { "Death": 1906 } }  ) |
| { "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 0 } |

1. Lägg till ytterligare författare i ”authors” genom att köra kommandot load(”addAuthors.js”) från mongo Shell. (Ladda ner filen från ITHS Distans).

|  |
| --- |
| load("D:\\Databas och adm. .NET20dec\\Vecka 20 - Dokumentdatabaser, MolnetAzure\\addAuthors.js") |
| true |

1. Räkna hur många dokument det finns totalt i ”authors”.

|  |
| --- |
| db.authors.find().count() |
| 6 |

1. Räkna hur många författare som heter August i förnamn.

|  |
| --- |
| db.authors.find({ "FirstName": "August" }).count() |
| 2 |

1. Lägg till Birth: 1858 och Death: 1940 för Selma Lagerlöf

|  |
| --- |
| db.authors.updateOne(  { "\_id": ObjectId("60acb3aa07db3fe8c8d672fa") },  { "$set": { "Birth": 1858, "Death": 1940 } }  ) |
| { "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 } |

1. Lägg till en array ”Books” för Selma Lagerlöf med följande böcker: ”Gösta Berlings saga”, ”En herrgårdssägen”, ”Nils Holgerssons underbara resa genom Sverige”

|  |
| --- |
| db.authors.update(  { "\_id": ObjectId("60acb3aa07db3fe8c8d672fa") },  { "$push": { "Books": { $each: [  "Gösta Berlings saga",  "En herrgårdssägen",  "Nils Holgerssons underbara resa genom Sverige"  ] } } }  ) |
| WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 }) |

1. Lägg till boken ”Vi på Saltkråkan” bland Astrid Lindgrens böcker.

|  |
| --- |
| db.authors.update(  { "\_id": ObjectId("60acbdccc038d5d0634310d5") },  { "$push": { "Books": "Vi på Saltkråkan" } }  ) |
| WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 }) |

1. Ta bort boken ”Bröderna Lejonhjärta” från Astrid Lindgrens böcker.

|  |
| --- |
| db.authors.update(  { "\_id": ObjectId("60acbdccc038d5d0634310d5") },  { "$pull": { "Books": "Bröderna Lejonhjärta" } }  ) |
| WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 }) |

1. Visa dokument för de författare som dog år 2000 eller senare.

|  |
| --- |
| db.authors.find({ "Death": { "$gte": 2000 } }).pretty() |
| {  "\_id" : ObjectId("60acbdccc038d5d0634310d5"),  "FirstName" : "Astrid",  "LastName" : "Lindgren",  "Birth" : 1907,  "Death" : 2002,  "Books" : [  "Här kommer Pippi Långstrump",  "Mio min Mio",  "Vi på Saltkråkan"  ]  } |

1. Räkna hur många författare som dog på 1940-talet.

|  |
| --- |
| db.authors.find({ "Death": { "$gte": 1940, "$lte": 1949 } }).count() |
| 3 |

1. Visa endast attributen FirstName, LastName och Death för de författare som dog på 1940-talet.

|  |
| --- |
| db.authors.find(  { "Death": { "$gte": 1940, "$lte": 1949 } },  { "FirstName": 1, "LastName": 1, "Death": 1, "\_id": 0 }  ).pretty() |
| { "FirstName" : "Selma", "LastName" : "Lagerlöf", "Death" : 1940 }  { "FirstName" : "Hjalmar", "LastName" : "Söderberg", "Death" : 1941 }  { "FirstName" : "Karin", "LastName" : "Boye", "Death" : 1941 } |

1. Lägg till Gender: ”Male” i dokument med FirstName = ”Hjalmar”.

|  |
| --- |
| db.authors.updateMany(  { "FirstName": "Hjalmar" },  { "$set": { "Gender": "Male" } }  ) |
| { "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 } |

1. Lägg till Gender: ”Female” i dokument där FirstName är Astrid, Karin eller Selma.

|  |
| --- |
| db.authors.updateMany(  { $or: [  { "FirstName": "Astrid" },  { "FirstName": "Karin" },  { "FirstName": "Selma" }  ] },  { "$set": { "Gender": "Female" } }  ) |
| { "acknowledged" : true, "matchedCount" : 3, "modifiedCount" : 3 } |

1. Ta bort arrayen Books från dokumentet med August Strindberg.

|  |
| --- |
| db.authors.update(  { "\_id": ObjectId("60acbdccc038d5d0634310d8") },  { "$unset": { "Books": "" } }  ) |
| WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 }) |

1. Ta bort dokumenten där FirstName = ”August”

|  |
| --- |
| db.authors.deleteMany( { "FirstName": "August" } ) |
| { "acknowledged" : true, "deletedCount" : 2 } |

Använd dig av kollektionen listingsAndReviews i databasen sample\_airbnb:

1. Ta ut (projicera) attributen name, price, och listing\_url för de dokument med property\_type: "House" och som har följande amenities: "Pool", "Air conditioning", "BBQ grill" och "Gym".

|  |
| --- |
| db.listingsAndReviews.find({  "property\_type": "House",  "amenities" : { "$all": [  "Pool",  "Air conditioning",  "BBQ grill",  "Gym"  ] } },  { "name": 1, "price": 1, "listing\_url": 1, "\_id": 0 })  .pretty() |
| {  "listing\_url" : "https://www.airbnb.com/rooms/24999091",  "name" : "Puamana Resort #20-2",  "price" : NumberDecimal("285.00")  }  {  "listing\_url" : "https://www.airbnb.com/rooms/6665504",  "name" : "Waikiki Banyan Tower 2 Suite 1214",  "price" : NumberDecimal("238.00")  }  {  "listing\_url" : "https://www.airbnb.com/rooms/16885508",  "name" : "Private Family Retreat with Spa & Pool",  "price" : NumberDecimal("249.00")  }  {  "listing\_url" : "https://www.airbnb.com/rooms/2271702",  "name" : "LUXURY HOUSE IN BARRA DA TIJUCA",  "price" : NumberDecimal("5502.00")  } |

Använd dig av kollektionen movies i databasen sample\_mflix:

1. Räkna antalet filmer där Drew Barrymore spelar mot Adam Sandler.

|  |
| --- |
| db.movies.find(  { "cast": { "$all": [  "Drew Barrymore",  "Adam Sandler" ] } }  ).count() |
| 3 |

1. Gruppera på attribut ”rated”, räkna antal filmer i varje grupp och sortera efter antal i fallande ordning (högst antal först).

|  |
| --- |
| db.movies.aggregate([  { "$project": { "rated": 1, "\_id": 0 } },  { "$group": { "\_id": "$rated", "count": { "$sum": 1 } } },  { $sort: { "count": -1 } }  ]) |
| { "\_id" : null, "count" : 9894 }  { "\_id" : "R", "count" : 5537 }  { "\_id" : "PG-13", "count" : 2321 }  { "\_id" : "PG", "count" : 1852 }  { "\_id" : "NOT RATED", "count" : 1354 }  { "\_id" : "UNRATED", "count" : 751 }  { "\_id" : "APPROVED", "count" : 709 }  { "\_id" : "G", "count" : 477 }  { "\_id" : "PASSED", "count" : 181 }  { "\_id" : "TV-14", "count" : 89 }  { "\_id" : "TV-PG", "count" : 76 }  { "\_id" : "TV-MA", "count" : 60 }  { "\_id" : "TV-G", "count" : 59 }  { "\_id" : "GP", "count" : 44 }  { "\_id" : "NC-17", "count" : 38 }  { "\_id" : "X", "count" : 38 }  { "\_id" : "M", "count" : 37 }  { "\_id" : "Approved", "count" : 5 }  { "\_id" : "AO", "count" : 3 }  { "\_id" : "TV-Y7", "count" : 3 }  { "\_id" : "Not Rated", "count" : 1 }  { "\_id" : "OPEN", "count" : 1 } |

1. Ta fram antalet filmer med Harrison Ford, och det lägsta, det högsta, samt genomsnittliga betyget bland dessa filmer på IMDB.

|  |
| --- |
| db.movies.aggregate([  { "$match": { "cast": { "$all": [ "Harrison Ford" ] } } },  { "$project": { "imdb.rating": 1, "\_id": 0 } },  { "$group": {  "\_id": "imdb.rating",  "count": { "$sum": 1 },  "min": { "$min": "$imdb.rating" },  "max": { "$max": "$imdb.rating" },  "avg": { "$avg": "$imdb.rating" } } }  ]) |
| { "\_id" : "imdb.rating", "count" : 25, "min" : 5.7, "max" : 8.8, "avg" : 7.127999999999999 } |

Använd dig av kollektionen sales i databasen sample\_supplies:

1. Skapa en vy (view) med namn satisfaction som visar genomsnittlig satisfaction per gender och purchaseMethod.

|  |
| --- |
| db.createView(  "satisfaction",  "sales",  [{ "$project": {  "customer.gender": 1,  "customer.satisfaction": 1,  "purchaseMethod": 1,  "\_id": 0 } },  { "$group": {  "\_id": {  "gender": "$customer.gender",  "purchaseMethod": "$purchaseMethod" },  "satisfaction": {  "$avg": "$customer.satisfaction" } } }  ]) |
| {  "ok" : 1,  "$clusterTime" : {  "clusterTime" : Timestamp(1622022504, 2),  "signature" : {  "hash" : BinData(0,"NlZuSPhAtWpD8UUeur/hUcpW3lo="),  "keyId" : NumberLong("6961782880305414147")  }  },  "operationTime" : Timestamp(1622022504, 2)  } |