You have a collection of movie ratings and the people who submitted the ratings; you use MongoDB Compass or the command line to investigate the data - and you realise that ratings are in a record with a key "`contractType`", and value "`Rating`".

It's now easy enough to search for all records pertaining to movie ratings …

```
> db.Data_Scientist_acs.find({"contractType": "Rating"})
{ "_id" : ObjectId("5c537510fe7bd25848132fcd"), "cid" : "#611:1", "contractType" : "Rating",
"payload" : { "person" : "p600", "analyst" : "Analyst", "rating" : { "id" : 2017, "rating" : 4,
"date" : ISODate("2009-03-22T08:46:14Z"), "name" : "Babes in Toyland", "year" : 1961, "genres" :
[ "Children", "Fantasy", "Musical" ] } } }
{ "_id" : ObjectId("5c537510fe7bd25848132fce"), "cid" : "#612:1", "contractType" : "Rating",
"payload" : { "person" : "p165", "analyst" : "Analyst", "rating" : { "id" : 5530, "rating" : 2,
"date" : ISODate("2003-03-04T01:30:11Z"), "name" : "Simone (S1m0ne)", "year" : 2002, "genres" :
[ "Comedy", "Drama", "Fantasy", "Sci-Fi" ] } } }
<snip>
```

… but you quickly realise there are just too many:

```
> db.Data_Scientist_acs.find({"contractType": "Rating"}).count()
100836
```

You can see some of the movie names (**Max** for example) and then drill down from there:

```
> db.Data_Scientist_acs.find({$and: [{"contractType": "Rating"}, {"payload.rating.name":
"Max"}]})
{ "_id" : ObjectId("5c537510fe7bd25848132fcf"), "cid" : "#613:1", "contractType" : "Rating",
"payload" : { "person" : "p165", "analyst" : "Analyst", "rating" : { "id" : 5993, "rating" : 2,
"date" : ISODate("2003-03-04T01:32:24Z"), "name" : "Max", "year" : 2002, "genres" : [ "Drama" ]
} } }
{ "_id" : ObjectId("5c537510fe7bd25848132fd1"), "cid" : "#615:1", "contractType" : "Rating",
"payload" : { "person" : "p182", "analyst" : "Analyst", "rating" : { "id" : 5993, "rating" : 3,
"date" : ISODate("2003-09-11T11:09:13Z"), "name" : "Max", "year" : 2002, "genres" : [ "Drama" ]
} } }
<snip>
```

But what if you didn't know the names of the movies available? Or even the fact that there were only two types of records? This is where MongoDB's **Aggregation Operations** come into their own; from MongoDB's own manual:

> "Aggregation operations process data records and return computed results. Aggregation operations group values from multiple documents together, and can perform a variety of operations on the grouped data to return a single result. MongoDB provides three ways to perform aggregation: the aggregation pipeline, the map-reduce function, and single purpose aggregation methods."

Let's look at Aggregation Pipelines, the preferred method for investigating data.

The Aggregation Pipeline:

In a nutshell - records in a collection go through a multi-stage pipeline that each stage perform query and filtering operations on the data, and also transforms the results into aggregated sets that input into the next stage of the pipeline - and so forth until the final result is arrived at. Other pipeline operations provide tools for grouping, sorting and counting records by specific fields, and also basic statistical functions like averages, summations, etc.

Let’s build a pipeline to investigate the data:

1. Let’s find all the contractTypes, and the number of contracts for each:

```
> db.Data_Scientist_acs.aggregate([
       { $group: { _id: "$contractType", count: { $sum: 1 } } }
])
{ "_id" : "Rating", "count" : 100836 }
{ "_id" : "Person", "count" : 610 }
```

2. Let’s look at only the `Person` records, and break it down by gender:

```
> db.Data_Scientist_acs.aggregate([
       { $match: { "contractType": "Main.MovieRatings:Person" }},
       { $group: { _id: "$payload.personInfo.gender", count: { $sum: 1 } } }
])
{ "_id" : { "Male" : {  } }, "count" : 292 }
{ "_id" : { "Female" : {  } }, "count" : 318 }
```

3. Let’s break it down by titles ...

```
> db.Data_Scientist_acs.aggregate([
       { $match: { "contractType": "Main.MovieRatings:Person" }},
       { $group: { _id: "$payload.personInfo.name.title", count: { $sum: 1 } } }
])
{ "_id" : { "Mrs" : {  } }, "count" : 118 }
{ "_id" : { "Ms" : {  } }, "count" : 91 }
{ "_id" : { "Mr" : {  } }, "count" : 292 }
{ "_id" : { "Miss" : {  } }, "count" : 109 }
```

4. And look for the top 5 most common last name …

```
> db.Data_Scientist_acs.aggregate([
       { $match: { "contractType": "Main.MovieRatings:Person" }},
       { $group: { _id: "$payload.personInfo.name.lastname", count: { $sum: 1 } } },
       { $sort: { count: -1 } },
       { $limit: 5 }
])
{ "_id" : "Ward", "count" : 8 }
{ "_id" : "Diaz", "count" : 8 }
{ "_id" : "Patterson", "count" : 7 }
{ "_id" : "Burke", "count" : 6 }
{ "_id" : "Gibson", "count" : 6 }
```

5. Let’s look at who the top rater was:

```
> db.Data_Scientist_acs.aggregate([
       { $match: { "contractType": "Main.MovieRatings:Rating" }},
       { $group: { _id: "$payload.person", count: { $sum: 1 }}},
       { $sort: { count: -1 } }
])
{ "_id" : "p414", "count" : 666 }
```

6. And their ratings breakdown:

```
> db.Data_Scientist_acs.aggregate([
       { $match: { $and: [
              { "contractType": "Main.MovieRatings:Rating" },
              { "payload.person": "p414"}
       ] } },
       { $group: { _id: "$payload.rating.rating", count: { $sum:1 } } },
       { $sort: { _id: -1 } }
])
{ "_id" : 5, "count" : 51 }
```

```
{ "_id" : 4.5, "count" : 20 }
{ "_id" : 4, "count" : 244 }
{ "_id" : 3.5, "count" : 69 }
{ "_id" : 3, "count" : 133 }
{ "_id" : 2.5, "count" : 36 }
{ "_id" : 2, "count" : 97 }
{ "_id" : 1.5, "count" : 7 }
{ "_id" : 1, "count" : 8 }
{ "_id" : 0.5, "count" : 1 }
```

7. Genres:

```
> db.Data_Scientist_acs.aggregate([
        { $match: { $and: [
                { "contractType": "Main.MovieRatings:Rating" },
                { "payload.person": "p414"}
        ] } },
        { $unwind: "$payload.rating.genres" },
        { $group: { _id: "$payload.rating.genres", count: { $sum:1 } } },
        { $sort: { count: -1 } }
])
{ "_id" : "Drama", "count" : 345 }
{ "_id" : "Comedy", "count" : 274 }
{ "_id" : "Action", "count" : 148 }
{ "_id" : "Thriller", "count" : 139 }
{ "_id" : "Romance", "count" : 119 }
{ "_id" : "Adventure", "count" : 98 }
{ "_id" : "Crime", "count" : 87 }
{ "_id" : "Sci-Fi", "count" : 72 }
{ "_id" : "Fantasy", "count" : 41 }
{ "_id" : "Children", "count" : 36 }
{ "_id" : "Musical", "count" : 32 }
{ "_id" : "War", "count" : 29 }
{ "_id" : "Mystery", "count" : 29 }
{ "_id" : "Animation", "count" : 25 }
{ "_id" : "Documentary", "count" : 25 }
{ "_id" : "Horror", "count" : 20 }
{ "_id" : "IMAX", "count" : 16 }
{ "_id" : "Western", "count" : 13 }
{ "_id" : "(no genres listed)", "count" : 1 }
{ "_id" : "Film-Noir", "count" : 1 }
```

MongoDB Data structures used in this tutorial:

Sample *Person* Record


```
{
  "_id" : ObjectId("5c53746bfe7bd25848132d7c"),
  "cid" : "#18:1",
  "contractType" : "Person",
  "payload" : {
    "analyst" : "Analyst",
    "person" : "p171",
    "personInfo" : {
      "name" : {
       "title" : {
         "Ms" : {  }
       },
       "firstname" : "Kimberly",
       "lastname" : "Herrera"
      },
      "gender" : {
       "Female" : {  }
      },
      "address" : {
       "street" : "1775 Windsor Road",
       "city" : "Salisbury",
       "state" : "Dumfries And Galloway",
       "postcode" : "J1 9ZW"
      },
      "email" : "kimberly.herrera@example.com",
      "dob" : ISODate("1976-09-09T11:38:37Z")
    }
  }
}
```


Sample *Rating* Record


```
{
  "_id" : ObjectId("5c537516fe7bd25848132fe0"),
  "cid" : "#630:1",
  "contractType" : "Rating",
  "payload" : {
    "person" : "p167",
    "analyst" : "Analyst",
    "rating" : {
      "id" : 6480,
      "rating" : 3,
      "date" : ISODate("2006-08-04T21:14:43Z"),
      "name" : "Thoroughly Modern Millie",
      "year" : 1967,
      "genres" : [
       "Comedy",
       "Musical"
      ]
    }
  }
}
```