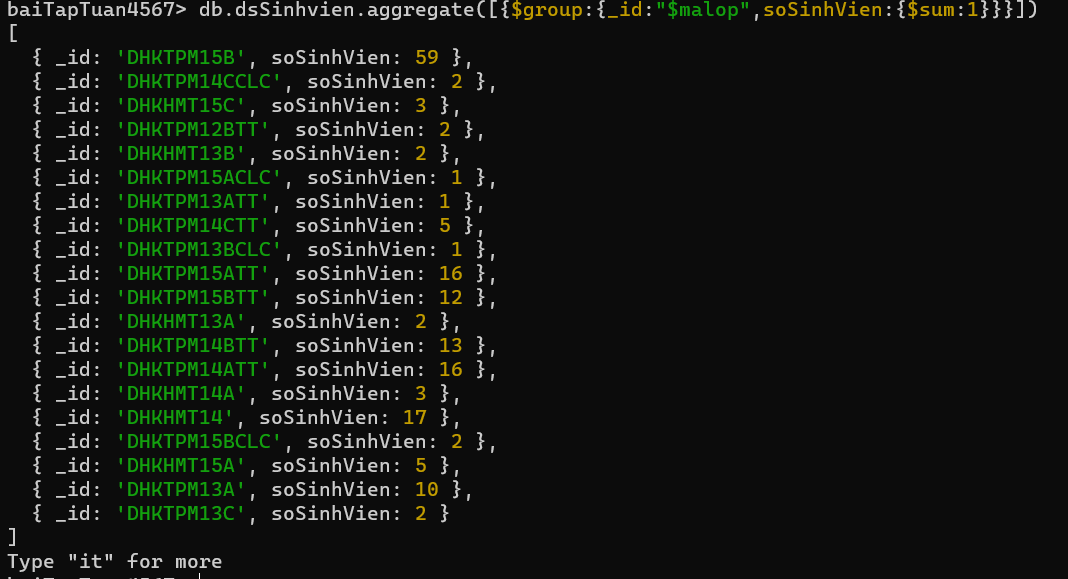
**Bài 1. Collection Sinhvien**

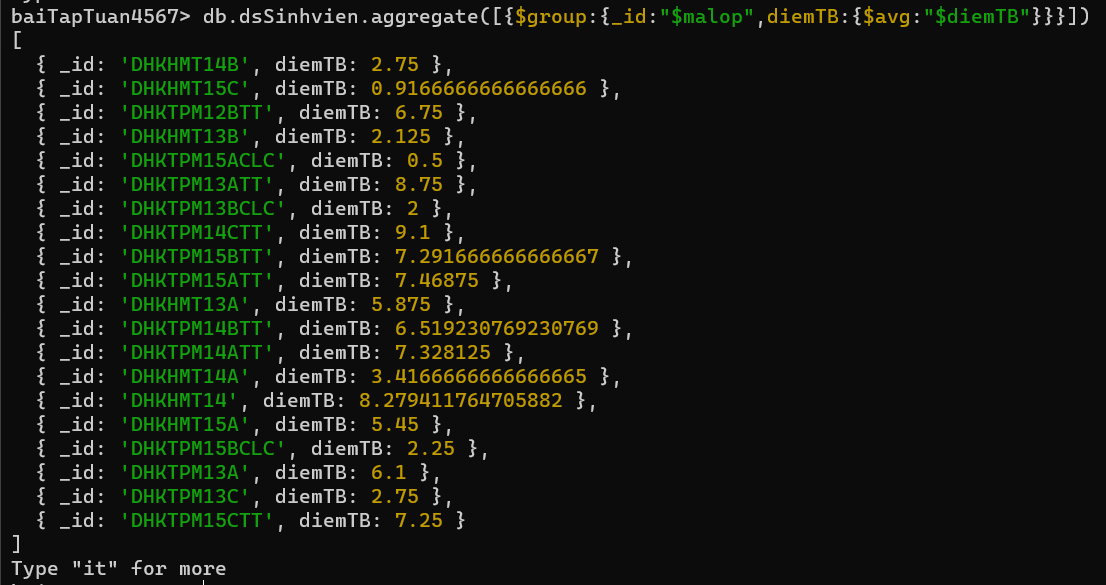
1. Đếm số sinh sinh của mỗi lớp, thông tin gồm mã lớp và số sinh viên.

db.dsSinhvien.aggregate([{$group:{\_id:"$malop",soSinhVien:{$sum:1}}}])



2. Tính điểm trung bình của từng lớp, thông tin gồm mã lớp và điểm trung bình.

db.dsSinhvien.aggregate([{$group:{\_id:"$malop",diemTB:{$avg:"$diemTB"}}}])



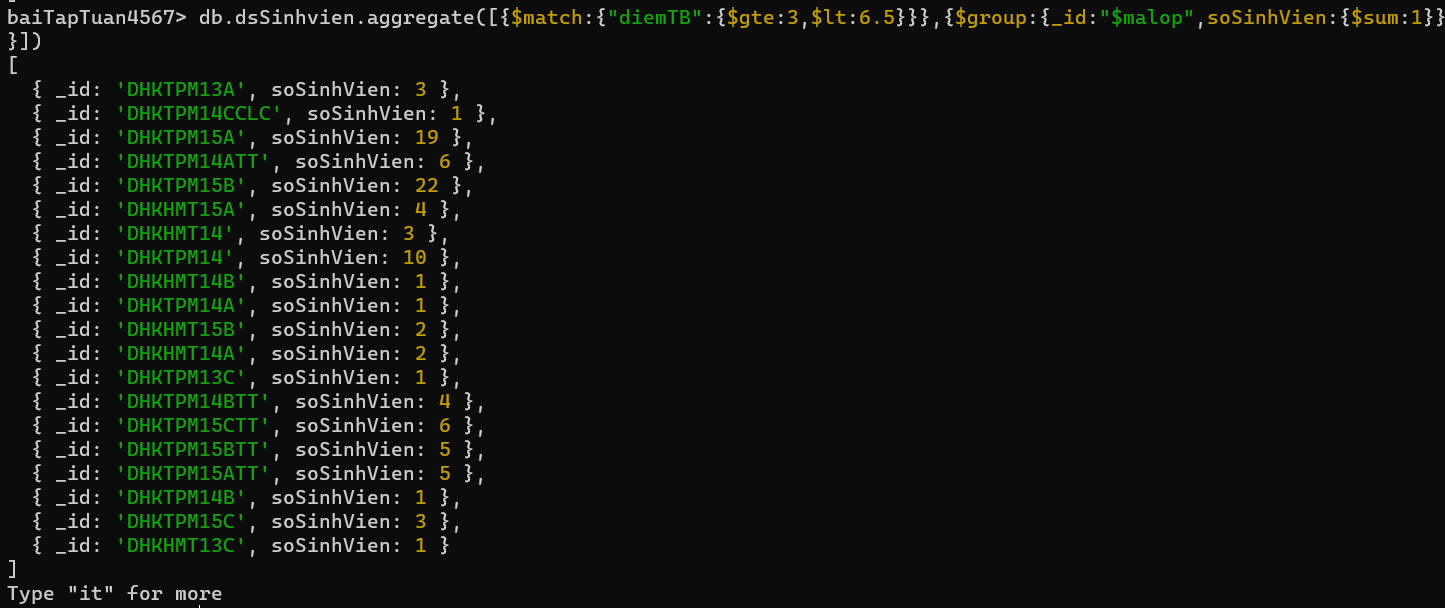
3. Đếm số sinh viên có điểm trung bình từ 5 trở lên của từng lớp, thông tin gồm mã lớp, số sinh viên.

db.dsSinhvien.aggregate([{$match:{"diemTB":{$gte:5}}},{$group:{\_id:"$malop",soSinhVien:{$sum:1}}}])



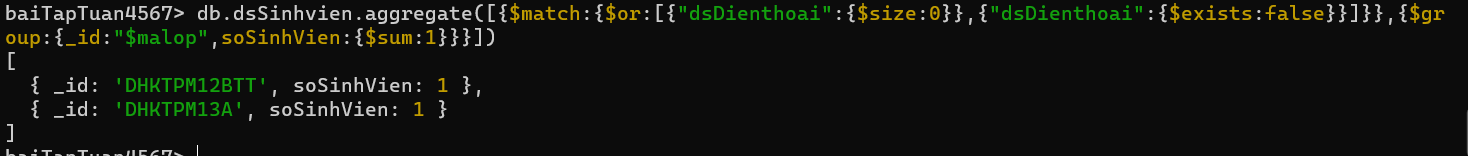
4. Đếm số sinh viên có điểm trung bình từ 3.0 đến dưới 6.5 của từng lớp, thông tin gồm mã lớp, số sinh viên.

db.dsSinhvien.aggregate([{$match:{"diemTB":{$gte:3,$lt:6.5}}},{$group:{\_id:"$malop",soSinhVien:{$sum:1}}}])



5. Đếm số sinh viên không có số điện thoại của từng lớp, thông tin gồm mã lớp, số sinh viên không điện thoại.

db.dsSinhvien.aggregate([{$match:{$or:[{"dsDienthoai":{$size:0}},{"dsDienthoai":{$exists:false}}]}},{$group:{\_id:"$malop",soSinhVien:{$sum:1}}}])

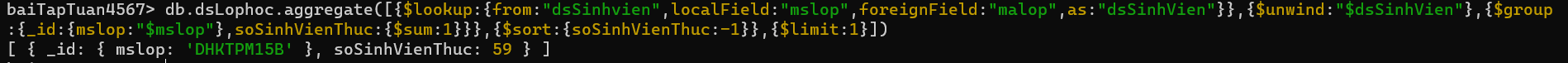


6. Đếm số sinh viên thực tế theo từng lớp, thông tin bao gồm thông tin của lớp học và tổng số sinh viên thực. (gồm 2 collection)

db.dsLophoc.aggregate([{$lookup:{from:"dsSinhvien",localField:"mslop",foreignField:"malop",as:"dsSinhVien"}},{$unwind:"$dsSinhVien"},{$group:{\_id:{macn:"$macn",mslop:"$mslop",sisoDukien:"$sisoDukien",tenlop:"$tenlop"},soSinhVienThuc:{$sum:1}}}])



7. Tìm lớp học có tổng số sinh viên thực tế cao nhất. Thông tin bao gồm thông tin của lớp học và tổng số sinh viên. [ { \_id: 'DHKTPM15B', soluong: 59 } ]

db.dsLophoc.aggregate([{$lookup:{from:"dsSinhvien",localField:"mslop",foreignField:"malop",as:"dsSinhVien"}},{$unwind:"$dsSinhVien"},{$group:{\_id:{mslop:"$mslop"},soSinhVienThuc:{$sum:1}}},{$sort:{soSinhVienThuc:-1}},{$limit:1}])

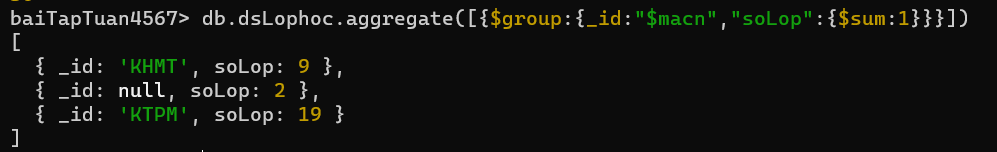
8. Đếm số sinh viên có điểm trung bình từ 9.0 trở lên theo từng lớp, sắp xếp theo tên sinh viên. Thông tin bao gồm thông tin của lớp học và tổng số sinh viên.

db.dsLophoc.aggregate([{$lookup:{from:"dsSinhvien",localField:"mslop",foreignField:"malop",as:"dsSinhVien"}},{$unwind:"$dsSinhVien"},{$match:{"dsSinhVien.diemTB":{$gte:9}}},{$group:{\_id:{macn:"$macn",mslop:"$mslop",sisoDukien:"$sisoDukien",tenlop:"$tenlop"},soSvCoDiemHon9:{$sum:1}}}])

9. Xuất danh sách sinh viên có điểm trung bình từ 9.0 trở lên theo từng lớp, sắp xếp theo tên sinh viên ra collection riêng biệt. Thông tin bao gồm thông tin của sinh viên.

10.Đếm số lớp theo chuyên ngành. Thông tin gồm mã chuyên ngành, số lớp

db.dsLophoc.aggregate([{$group:{\_id:"$macn","soLop":{$sum:1}}}])



11.Tính tổng số sinh viên theo từng chuyên ngành. Thông tin bao gồm thông tin của chuyên ngành và tổng số sinh viên.

db.dsLophoc.aggregate([{$lookup:{from:"dsSinhvien",localField:"mslop",foreignField:"malop",as:"dsSinhVien"}},{$unwind:"$dsSinhVien"},{$group:{\_id:{macn:"$macn",mslop:"$mslop",sisoDukien:"$sisoDukien",tenlop:"$tenlop"},soSVTheoLop:{$sum:1}}},{$group:{\_id:"$\_id.macn",soSinhVienTheoCN:{$sum:"$soSVTheoLop"}}}])



**Bài 2. Collection customers**

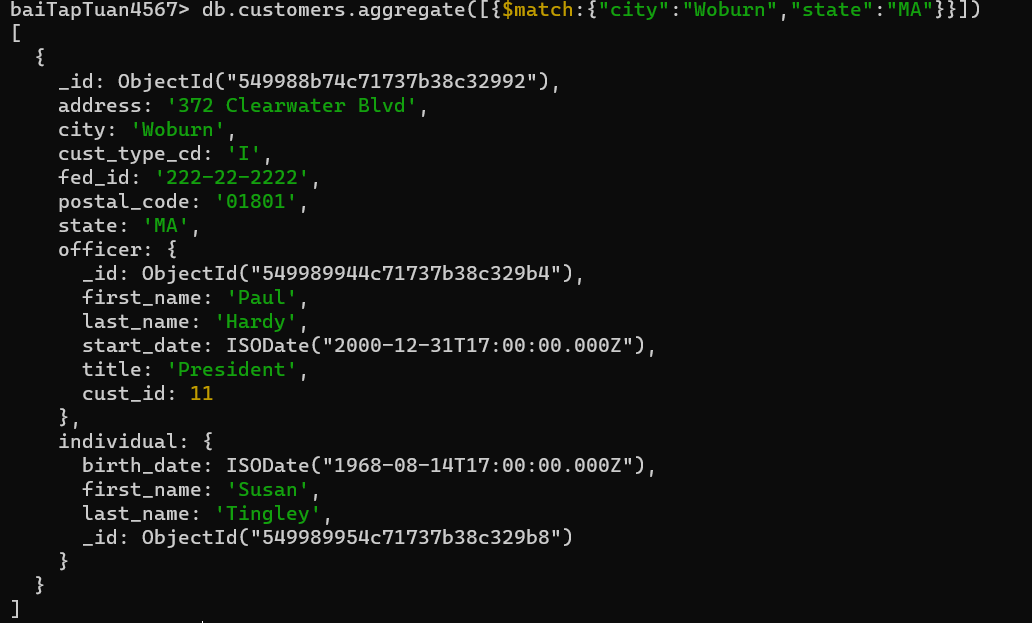
1. Xuất danh sách khách hàng với các thông tin address, city, state.

db.customers.aggregate([{$project:{address:1,city:1,state:1,\_id:0}}])



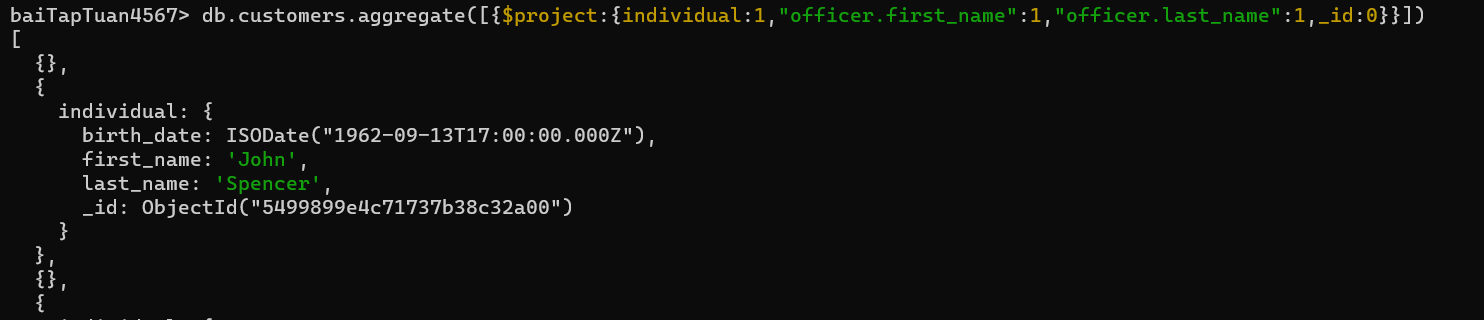
2. Xuất các khách hàng có city ở “Woburn” và state là “MA”.

db.customers.aggregate([{$match:{"city":"Woburn","state":"MA"}}])



3. Xuất thông tin khách hàng bao gồm individual, first\_name, last\_name của officer.

db.customers.aggregate([{$project:{individual:1,"officer.first\_name":1,"officer.last\_name":1,\_id:0}}])



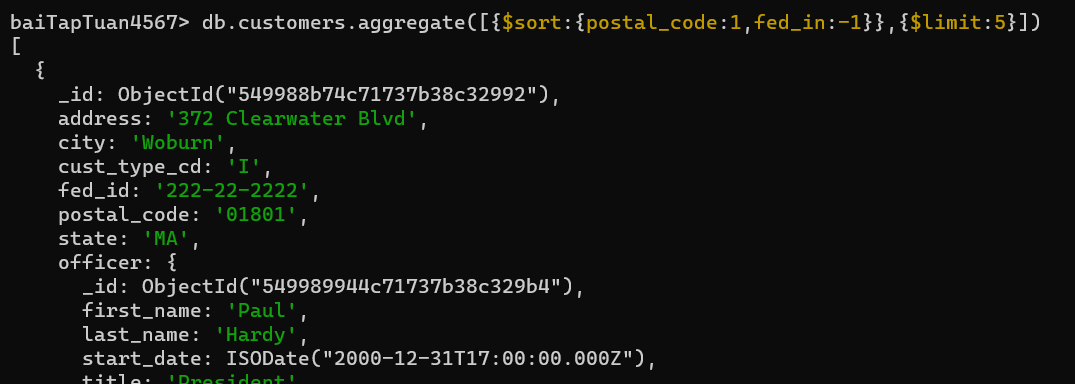
4. Nhóm danh sách khách hàng theo city và cho biết số lượng khách hàng tương ứng với điều kiện là các khách hàng có city ở “Salem”.

db.customers.aggregate([{$group:{\_id:"$city",soKhachHang:{$sum:1}}},{$match:{\_id:"Salem"}}])



5. Xuất danh sách 5 khách hàng đầu tiên được sắp xếp theo thứ tự tăng dần của postal\_code và giảm dần của fed\_id

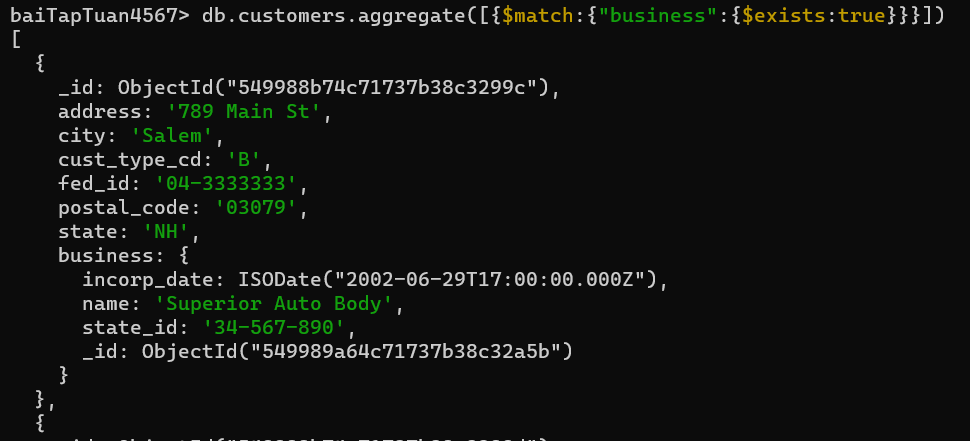
db.customers.aggregate([{$sort:{postal\_code:1,fed\_in:-1}},{$limit:5}])



6. Lọc các khách hàng có state là “MA”. Sau đó, thực hiện nhóm các khách hàng vừa tìm được theo city và đếm số lượng khách hàng tương ứng với danh sách city vừa nhóm. Tiếp theo thực hiện xuất 03 dữ liệu cuối cùng được sắp xếp kết quả giảm dần theo postal\_code. \*\*\*{city}

7. Xuất danh sách khách hàng có tồn tại field business.

db.customers.aggregate([{$match:{"business":{$exists:true}}}])



8. Xuất danh sách khách hàng có tồn tại field business với các thông tin fed\_id, cust\_type\_cd, state, khuyenMai. Trong đó field khuyenMai được tính như sau: nếu field state = “MA” thì sẽ được hưởng khuyến mãi (khuyenMai: true), ngược lại sẽ không được hưởng khuyến mãi (khuyenMai: false). KM:{ $cond: { if: {state:”MA”}, then: “giam 10%”, else: “khong km” } }

db.customers.aggregate([{$match:{"business":{$exists:true}}},{$project:{fed\_id:1,cust\_type\_cd:1,state:1,\_id:0,khuyenMai:{$cond:{if:{state:'MA'},then:true,else:false}}}}])

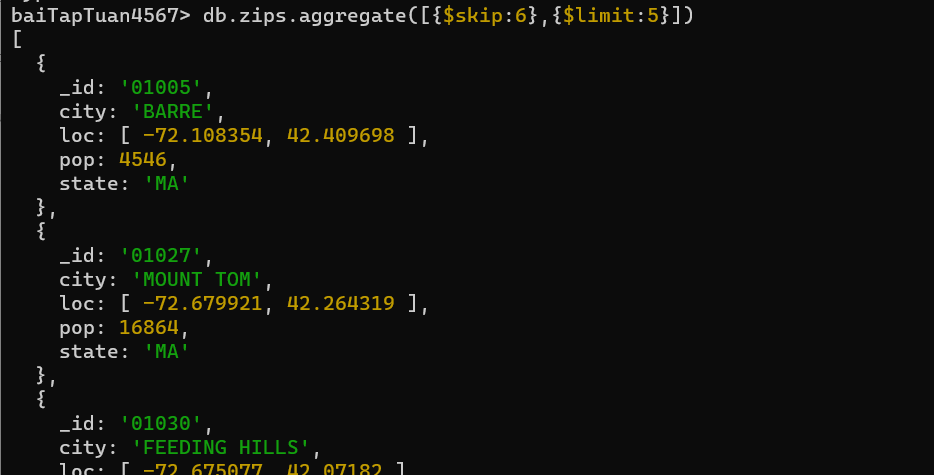
9. Nếu field khuyenMai là true (câu 8) thì sẽ hiển thị là “Được giảm 10%”, ngược lại sẽ hiển thị “Không được hưởng khuyến mãi”.

**Bài 3. Collection zips**

1. Hiển thị n documents từ document thứ k. (n, k tùy ý)

n = 5 k = 7

db.zips.aggregate([{$skip:6},{$limit:5}])



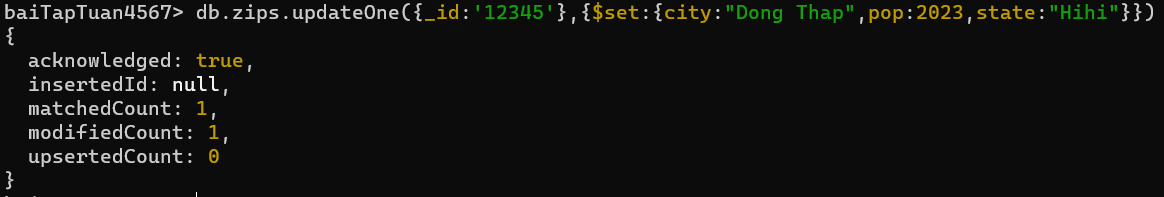
2. Chèn thêm 1 document mới. (tùy ý)

db.zips.insertOne({\_id:'12345',city:"TP HCM",loc:[-70.123454,40.454321],pop:2003,state:'Haha'})



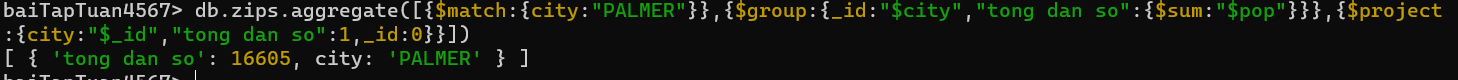
3. Cập nhật thông tin của một document khi biết id bất kỳ.

db.zips.updateOne({\_id:'12345'},{$set:{city:"Dong Thap",pop:2023,state:"Hihi"}})



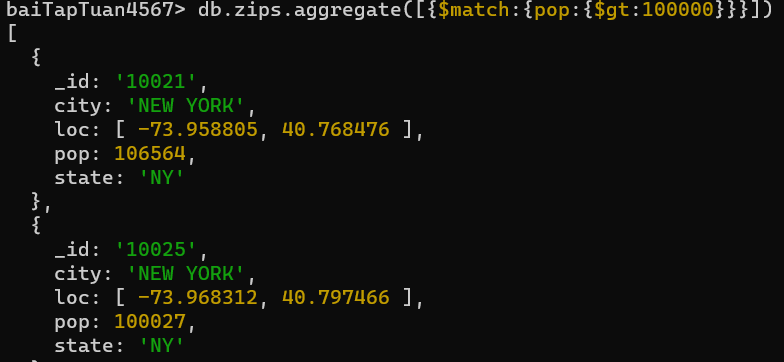
4. Tìm dân số của thành phố (city) PALMER.

db.zips.aggregate([{$match:{city:"PALMER"}},{$group:{\_id:"$city","tong dan so":{$sum:"$pop"}}},{$project:{city:"$\_id","tong dan so":1,\_id:0}}])



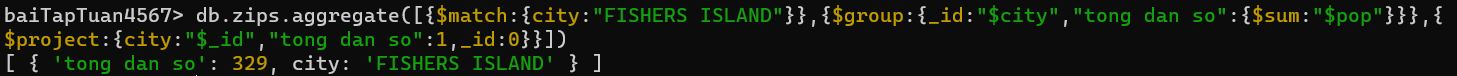
5. Tìm các document có dân số (pop) >100000

db.zips.aggregate([{$match:{pop:{$gt:100000}}}])

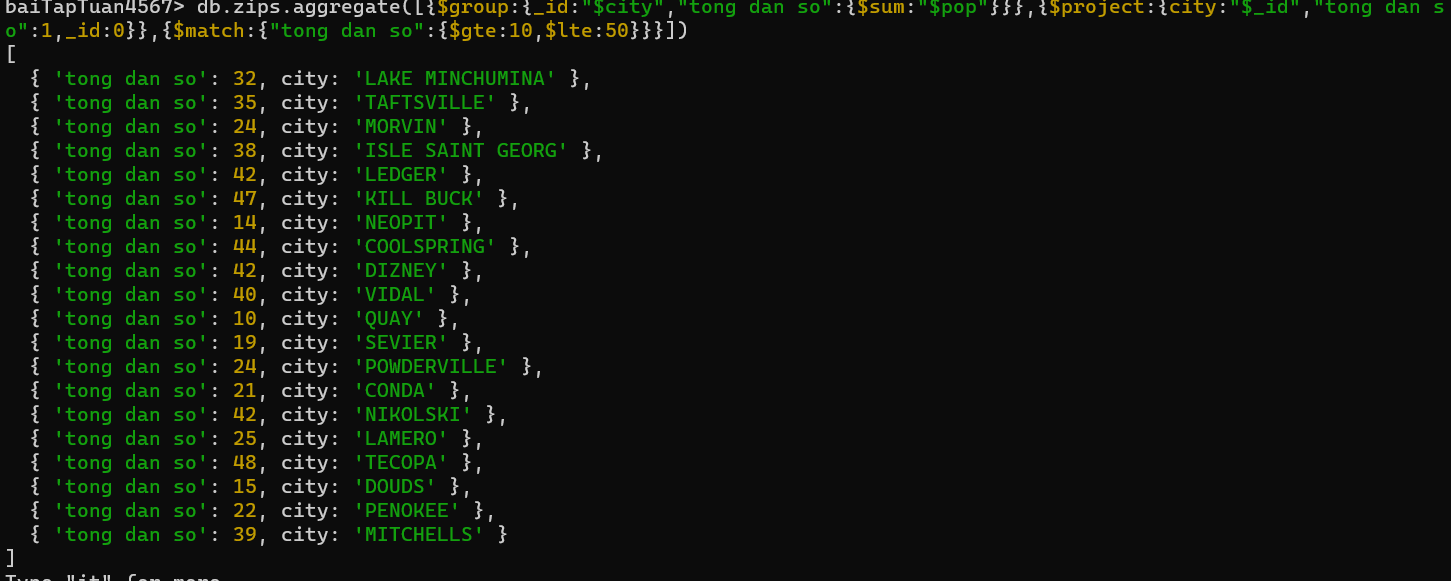


6. Tìm dân số của thành phố (city) FISHERS ISLAND

db.zips.aggregate([{$match:{city:"FISHERS ISLAND"}},{$group:{\_id:"$city","tong dan so":{$sum:"$pop"}}},{$project:{city:"$\_id","tong dan so":1,\_id:0}}])



7. Tìm các thành phố có dân số từ 10 – 50

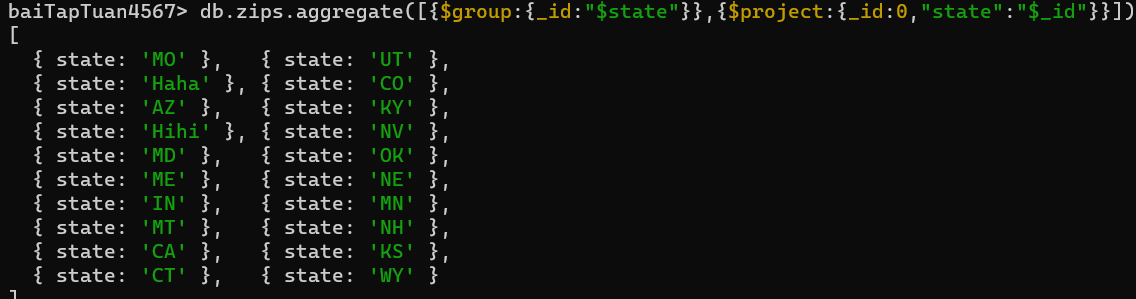


8. Tìm tất cả các thành phố của bang MA có dân số trên 500

db.zips.aggregate([{$group:{\_id:{state:"$state",city:"$city"},"tong dan so":{$sum:"$pop"}}},{$match:{"\_id.state":'MA',"tong dan so":{$gt:500}}}])

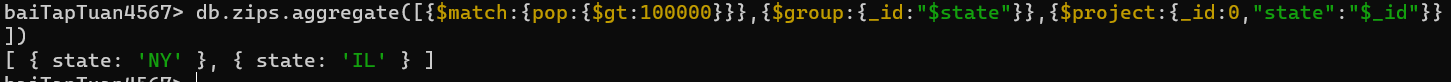


9. Tìm tất cả các bang (không trùng) db.zips.aggregate([{$group:{\_id:"$state"}},{$project:{\_id:0,"state":"$\_id"}}])



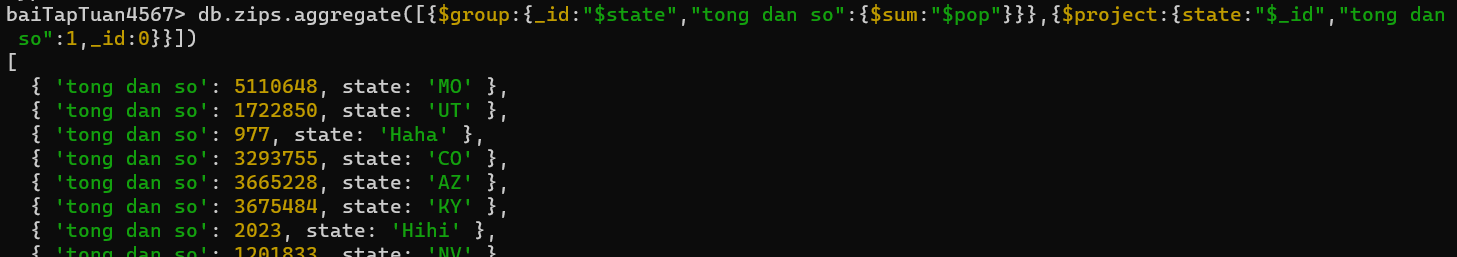
10.Tìm tất cả các bang mà có chứa ít nhất 1 thành phố có dân số trên 100000

db.zips.aggregate([{$match:{pop:{$gt:100000}}},{$group:{\_id:"$state"}},{$project:{\_id:0,"state":"$\_id"}}])



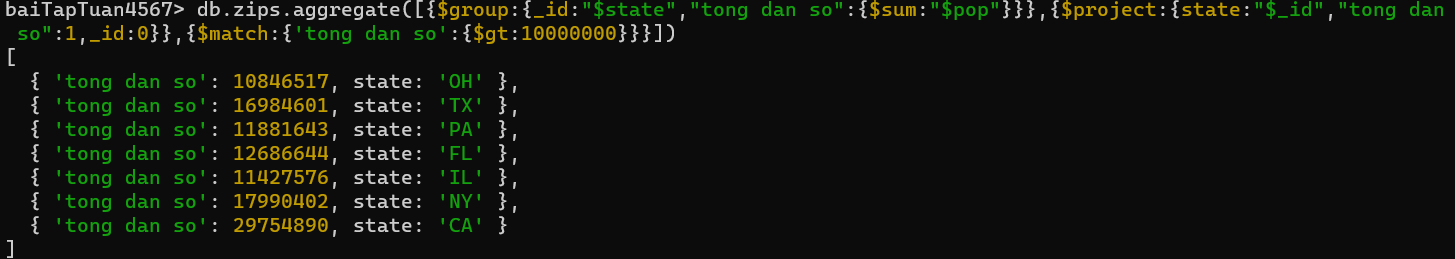
11.Tính tổng số dân (pop) theo từng bang (state).

db.zips.aggregate([{$group:{\_id:"$state","tong dan so":{$sum:"$pop"}}},{$project:{state:"$\_id","tong dan so":1,\_id:0}}])



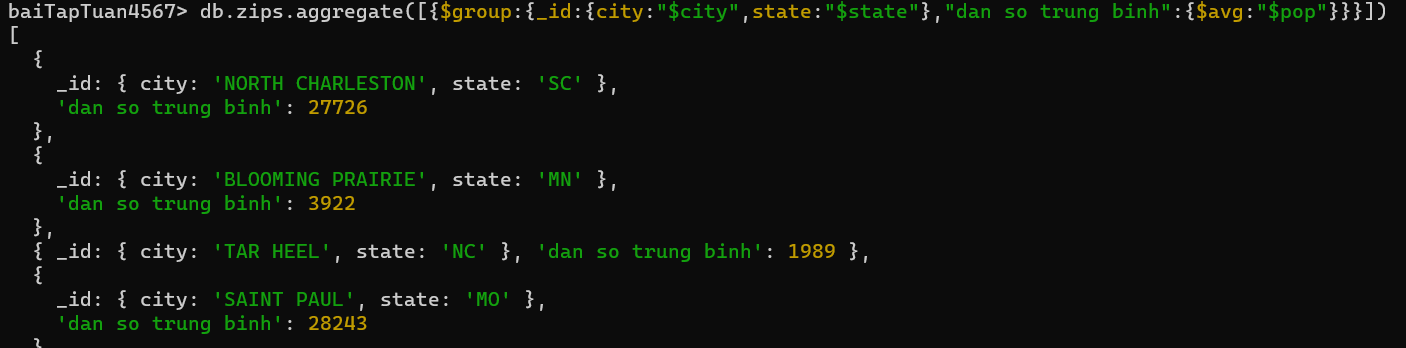
12.Tìm tất cả các bang có tổng dân số trên 10.000.000

db.zips.aggregate([{$group:{\_id:"$state","tong dan so":{$sum:"$pop"}}},{$project:{state:"$\_id","tong dan so":1,\_id:0}},{$match:{'tong dan so':{$gt:10000000}}}])



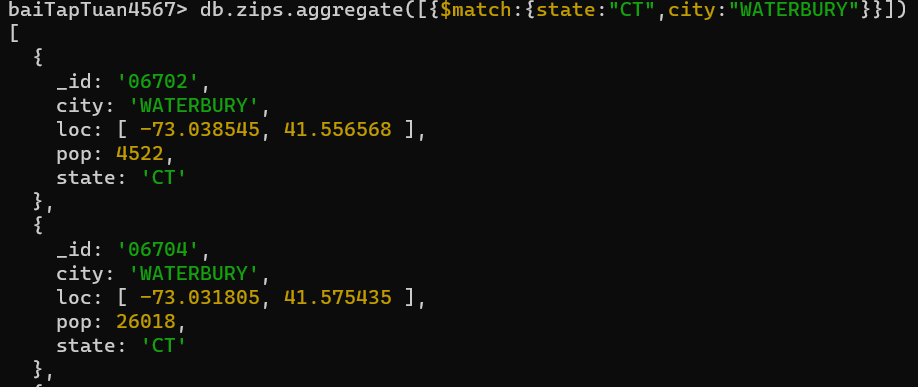
13.Tính dân số trung bình (các thành phố) theo từng bang (state).

db.zips.aggregate([{$group:{\_id:{city:"$city",state:"$state"},"dan so trung binh":{$avg:"$pop"}}}])



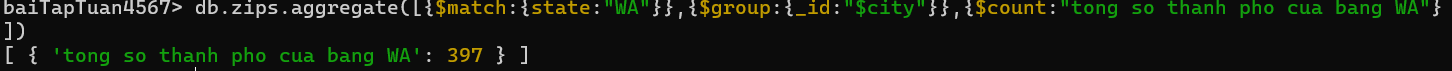
14.Tìm những document của bang 'CT' và thành phố 'WATERBURY'

db.zips.aggregate([{$match:{state:"CT",city:"WATERBURY"}}])



15.Bang WA có bao nhiêu city (nếu trùng chỉ tính 1 lần)\*\*

db.zips.aggregate([{$match:{state:"WA"}},{$group:{\_id:"$city"}},{$count:"tong so thanh pho cua bang WA"}])



16.Tính số city của mỗi bang (nếu trùng chỉ tính 1 lần), kết quả giảm dần theo số city

db.zips.aggregate([{$group:{\_id:{state:"$state",city:"$city"}}},{$group:{\_id:"$\_id.state","tong so city":{$sum:1}}}])

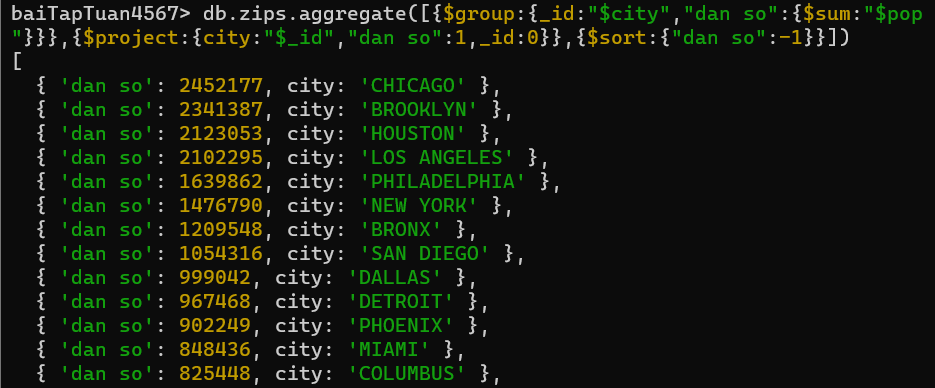


17.Group theo state va city, sum:1; Group state, sum

18.Tìm ra các thành phố có dân số (pop) lớn (nhỏ) nhất.

+ Các thành phố có dân số lớn nhất

db.zips.aggregate([{$group:{\_id:"$city","dan so":{$sum:"$pop"}}},{$project:{city:"$\_id","dan so":1,\_id:0}},{$sort:{"dan so":-1}}])



+ Các thành phố có dân số nhỏ nhất

db.zips.aggregate([{$group:{\_id:"$city","dan so":{$sum:"$pop"}}},{$project:{city:"$\_id","dan so":1,\_id:0}},{$sort:{"dan so":1}}])

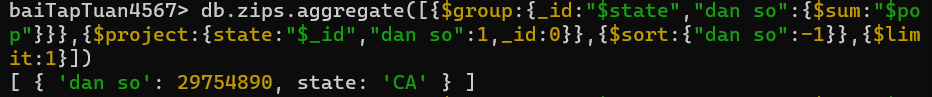


19.Sort theo pop, $groupΛ)

20.Tìm bang có dân số (pop) lớn (nhỏ) nhất.

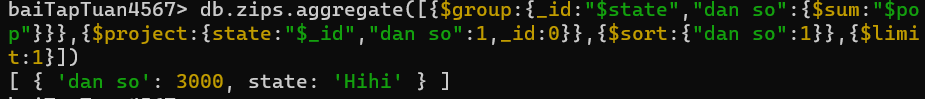
+ Bang có dân số lớn nhất

db.zips.aggregate([{$group:{\_id:"$state","dan so":{$sum:"$pop"}}},{$project:{state:"$\_id","dan so":1,\_id:0}},{$sort:{"dan so":-1}},{$limit:1}])



+ Bang có dân số lớn nhất

db.zips.aggregate([{$group:{\_id:"$state","dan so":{$sum:"$pop"}}},{$project:{state:"$\_id","dan so":1,\_id:0}},{$sort:{"dan so":1}},{$limit:1}])

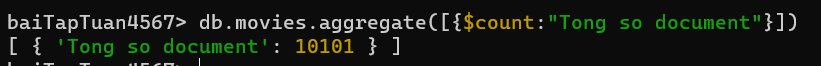


21.Xuất những document có dân số dưới dân số trung bình của mỗi city

**Bài 4. Collection movies**

1. Đếm tổng số các document movies.

db.movies.aggregate([{$count:"Tong so document"}])



2. Xuất các document movies theo năm, tính tổng số film trong mỗi năm

db.movies.aggregate([{$group:{\_id:"$year","Tong so film":{$sum:1}}},{$project:{\_id:0,"year":"$\_id","Tong so film":1}}])



3. Xuất các document movies theo năm, tính tổng số film trong mỗi năm, sau đó sắp xếp tăng.

db.movies.aggregate([{$group:{\_id:"$year","Tong so film":{$sum:1}}},{$project:{\_id:0,"year":"$\_id","Tong so film":1}},{$sort:{year:1}}])



4. Xuất các document movies theo năm, sau đó sắp xếp theo thứ tự giảm dần dựa trên số lượng.

db.movies.aggregate([{$group:{\_id:"$year","Tong so film":{$sum:1}}},{$project:{\_id:0,"year":"$\_id","Tong so film":1}},{$sort:{"Tong so film":-1}}])



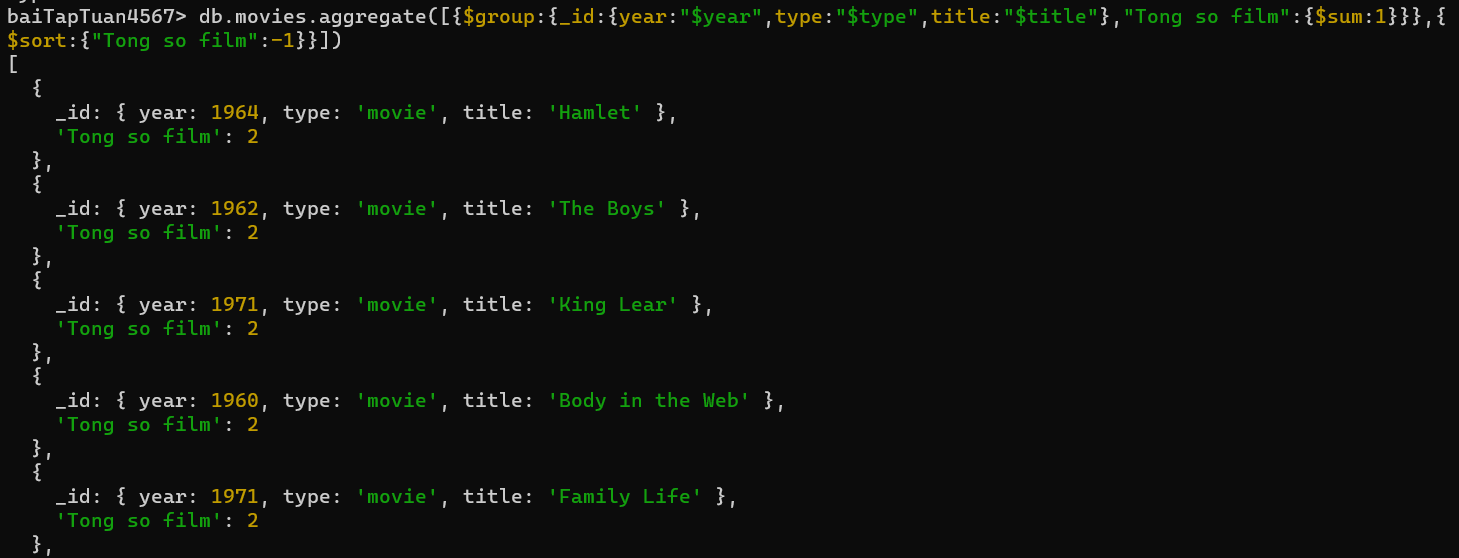
5. Xuất các document movies theo số lượng film mỗi đạo diễn có được,

db.movies.aggregate([{$unwind:"$directors"},{$group:{\_id:"$directors","so film co duoc":{$sum:1}}},{$project:{\_id:0,director:"$\_id","so film co duoc":1}}])



6. Xuất các document movies theo số lượng film từng year, type, title. Sau đó sắp xếp giảm dần theo số lượng đếm được.

db.movies.aggregate([{$group:{\_id:{year:"$year",type:"$type",title:"$title"},"Tong so film":{$sum:1}}},{$sort:{"Tong so film":-1}}])



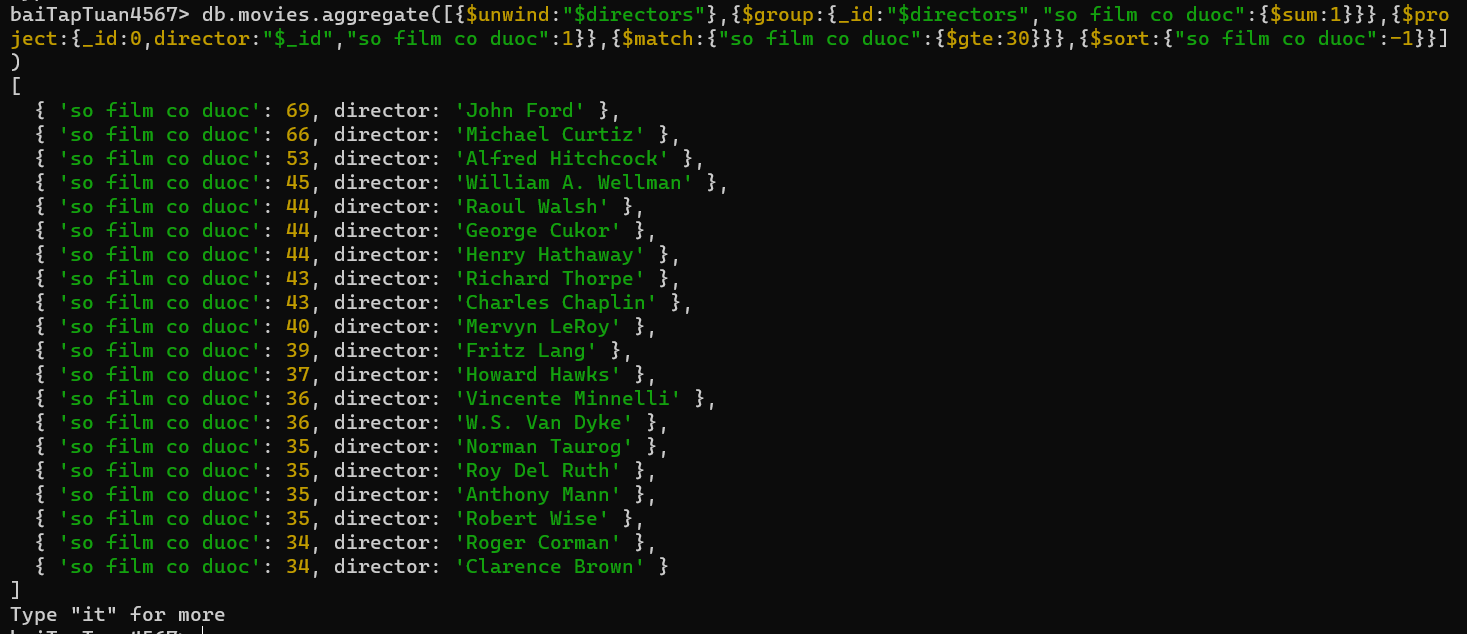
7. Liệt kê danh sách các đạo diễn có tham gia từ 30 bộ phim trở lên. Thông tin bao gồm: Tên đạo diễn (director) và số bộ phim.

db.movies.aggregate([{$unwind:"$directors"},{$group:{\_id:"$directors","so film co duoc":{$sum:1}}},{$project:{\_id:0,director:"$\_id","so film co duoc":1}},{$match:{"so film co duoc":{$gte:30}}}])



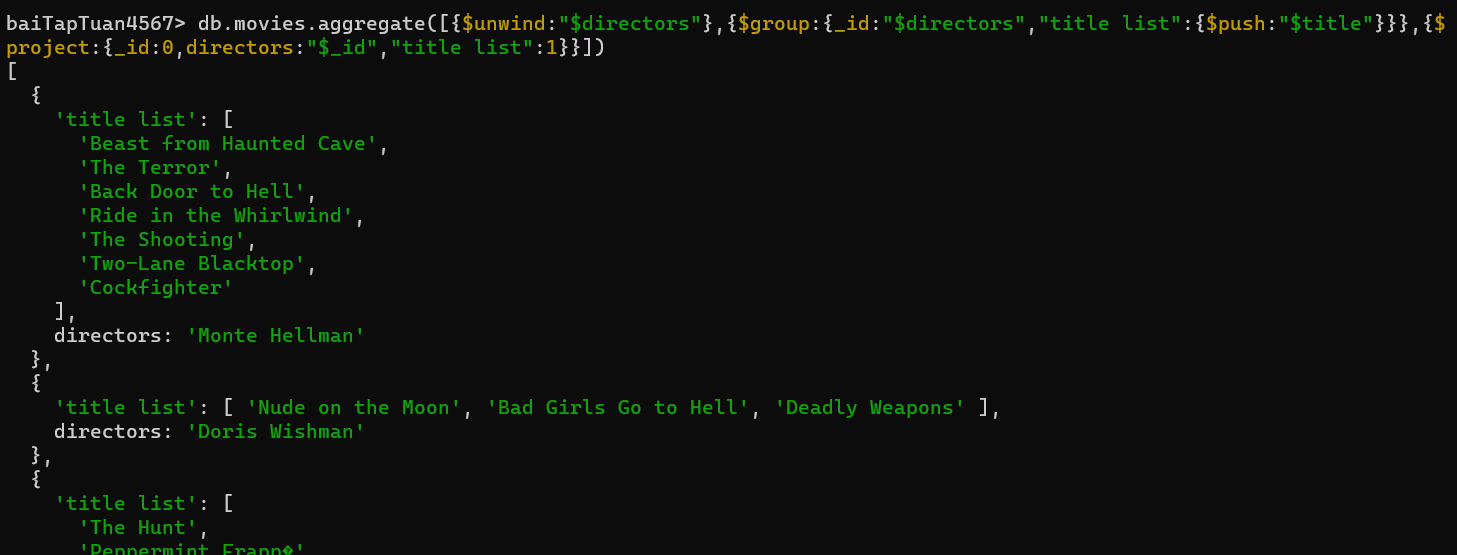
8. Tìm tất cả các đạo diễn có tham gia đạo diễn nhiều bộ phim nhất

db.movies.aggregate([{$unwind:"$directors"},{$group:{\_id:"$directors","so film co duoc":{$sum:1}}},{$project:{\_id:0,director:"$\_id","so film co duoc":1}},{$match:{"so film co duoc":{$gte:30}}},{$sort:{"so film co duoc":-1}}])



9. Liệt kê tựa phim (title) theo từng đạo diễn. Thông tin bao gồm: tên đạo diễn (director) và danh sách tựa phim

db.movies.aggregate([{$unwind:"$directors"},{$group:{\_id:"$directors","title list":{$push:"$title"}}},{$project:{\_id:0,directors:"$\_id","title list":1}}])



10.Thống kê số bộ phim đã phát hành theo từng năm, sắp xếp giảm dần theo năm

11.Tìm năm phát hành nhiều bộ phim nhất.

12.Liệt kê danh sách các tựa phim (title) theo từng quốc gia. Thông tin bao gồm: tên quốc gia và danh sách tựa phim

13.Đếm số bộ phim theo từng quốc gia, sắp xếp giảm dần theo số bộ phim. Thông tin bao gồm:Tên quốc gia và số bộ phim

14.Tìm những tựa phim (title) phát hành trong tháng 03 năm 2016

15.Liệt kê những tựa phim (title) do diễn viên “Frank Powell” hoặc “Charles Wellesley” đóng

16.Tìm những quốc gia phát hành nhiều bộ phim nhất