MongoDB: Advanced Queries (Aggregation Framework)

Prepared by Jeong-Hun Kim

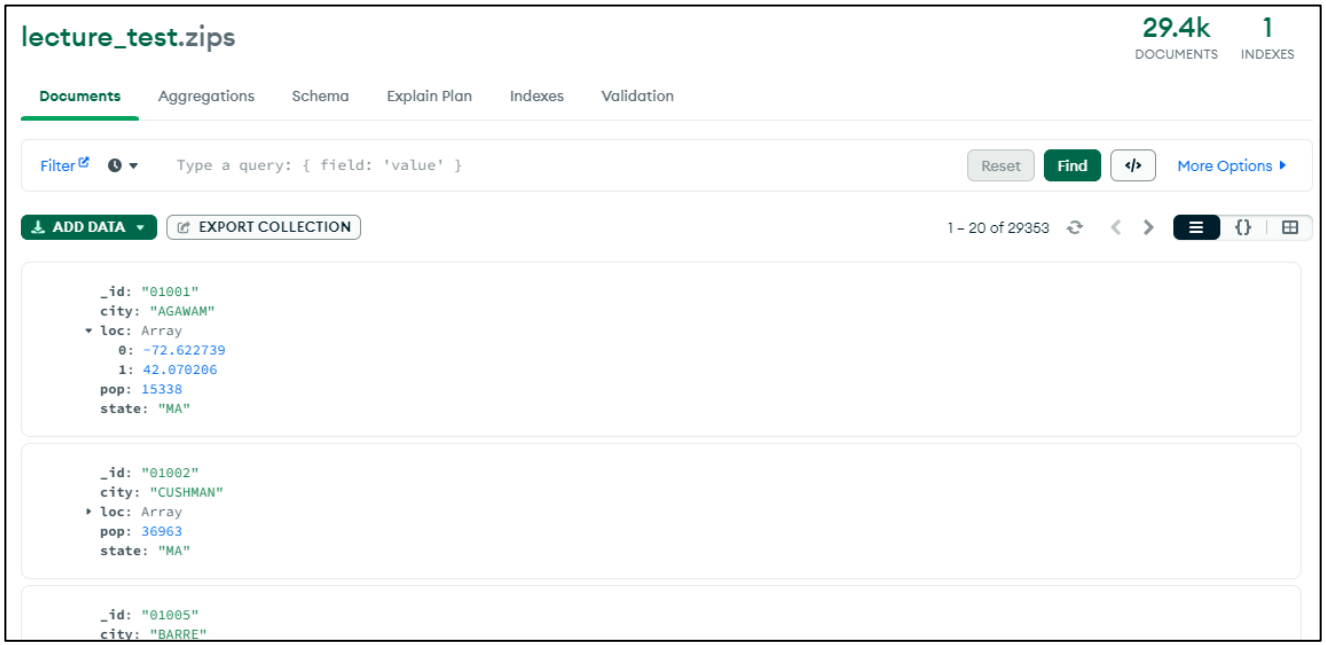
Table of Content

- In the last lecture
- Aggregation Framework
- Aggregation Pipeline
- Aggregation Pipeline Stages
- **Practice**

Practice

- Download the dataset
  - http://media.mongodb.org/zips.json

- Import zips database using MongoDB Compass



Practice

- $group
  - *{ $group: { _id: <expression>, <field1>: { <accumulator1> : <expression1> }, ... } }*

- Q1: State-wise total population

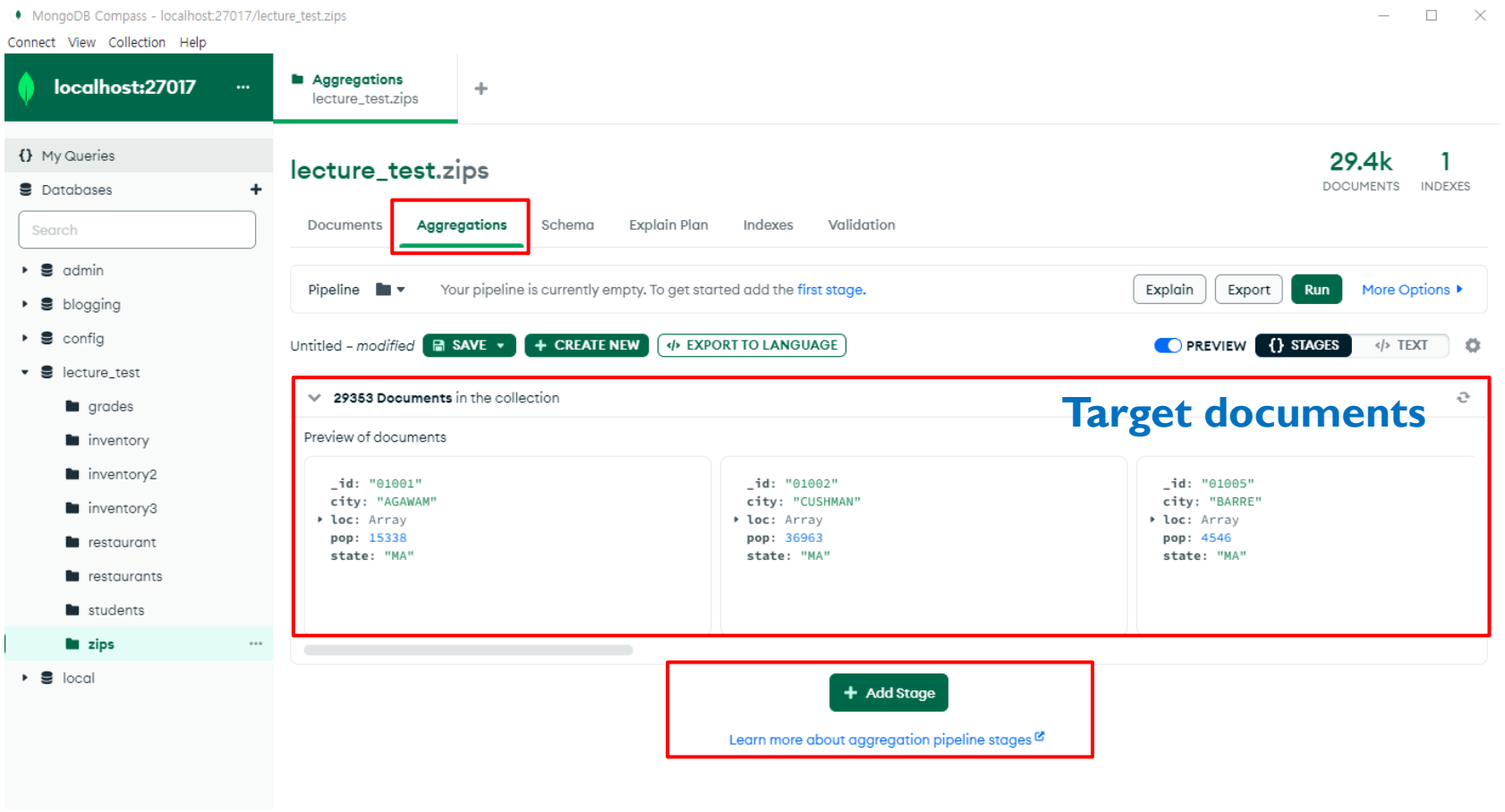
```
db.zips.aggregate ({$group:
  {
    _id: '$state',
    population: {$sum: '$pop'}
  }
})
```

See result

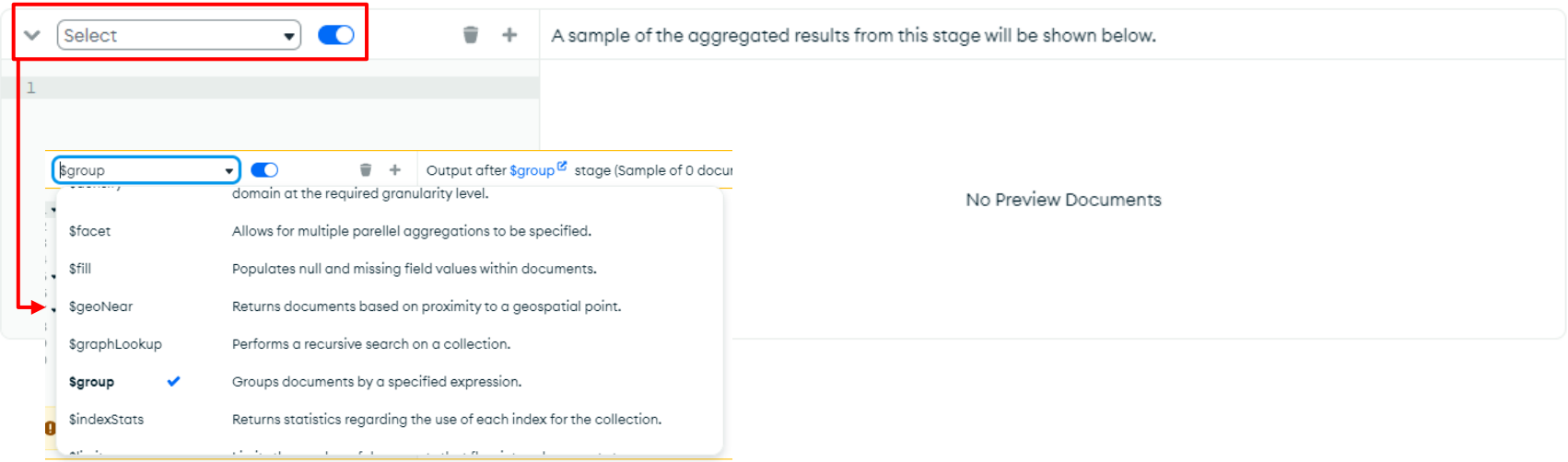
Practice

```
>_MONGOSH
< {
    _id: 'AL',
    population: 4040587
  }
  {
    _id: 'TN',
    population: 4876457
  }
  {
    _id: 'KS',
    population: 2475285
  }
  {
    _id: 'NE',
    population: 1578139
  }
  {
    _id: 'ID',
    population: 1006749
  }
  {
    _id: 'HI',
    population: 1108229
  }
  {
    _id: 'KY',
    population: 3675484
  }
```

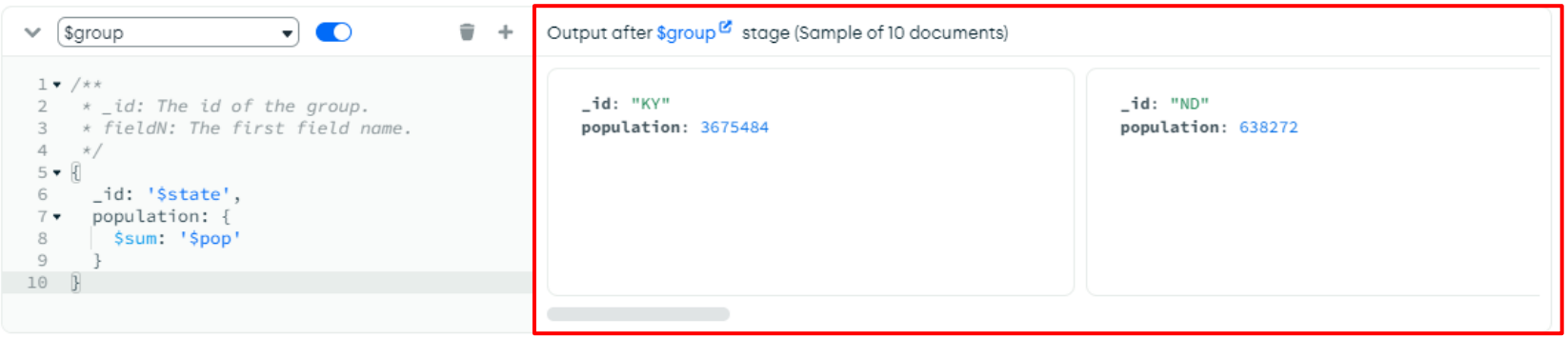
Practice


MongoDB Compass - localhost:27017/lecture_test.zips
Connect View Collection Help
localhost:27017
Aggregations
lecture_test.zips
My Queries
Databases
Search
admin
blogging
config
lecture_test
grades
inventory
inventory2
inventory3
restaurant
restaurants
students
zips
local
lecture_test.zips
29.4k
DOCUMENTS
1
INDEXES
Documents
Aggregations
Schema
Explain Plan
Indexes
Validation
Pipeline
Your pipeline is currently empty. To get started add the first stage.
Explain
Export
Run
More Options
Untitled – modified
SAVE
CREATE NEW
EXPORT TO LANGUAGE
PREVIEW
STAGES
TEXT
29353 Documents in the collection
Target documents
Preview of documents
_id: "01001"
city: "AGAWAM"
loc: Array
pop: 15338
state: "MA"
_id: "01002"
city: "CUSHMAN"
loc: Array
pop: 36963
state: "MA"
_id: "01005"
city: "BARRE"
loc: Array
pop: 4546
state: "MA"
Add Stage
Learn more about aggregation pipeline stages

Practice



Results (next stage target documents)



Practice

- $group
  - { $group: { _id: <expression>, <field1>: { <accumulator1> : <expression1> }, ... } }

- Q2: Count of zipcodes for states

```
db.zips.aggregate ({$group:
  {
    _id: '$state',
    population: {$sum: '$pop'},
    zips: {$sum: 1}
  }
})
```

See result

New field is included to count of zipcodes

Practice

```
< {
    _id: 'DE',
    population: 666168,
    zips: 53
  }
  {
    _id: 'TN',
    population: 4876457,
    zips: 582
  }
  {
    _id: 'KS',
    population: 2475285,
    zips: 715
  }
  {
    _id: 'NE',
    population: 1578139,
    zips: 574
  }
  {
    _id: 'ID',
    population: 1006749,
    zips: 244
  }
  {
    _id: 'HI',
    population: 1108229,
    zips: 80
  }
  {
    _id: 'KY',
    population: 3675484,
```

Practice

- $sort
  - { $sort: { <field1>: <sort order>, <field2>: <sort order> ... } }

- Q3: Sort cities by number of zips

```
db.zips.aggregate ({$group:
  {
    _id: '$state',
    population: {$sum: '$pop'},
     zips: {$sum: 1}
  }
},
{$sort:
    {zips: -1}
  }
)
```

See result

1: ascending
-1: descending

New stage is included to sort the zipcodes

Practice

```
< {
    _id: 'TX',
    population: 16984601,
    zips: 1671
  }
  {
    _id: 'NY',
    population: 17990402,
    zips: 1595
  }
  {
    _id: 'CA',
    population: 29754890,
    zips: 1516
  }
  {
    _id: 'PA',
    population: 11881643,
    zips: 1458
  }
  {
    _id: 'IL',
    population: 11427576,
    zips: 1237
  }
  {
    _id: 'OH',
    population: 10846517,
    zips: 1007
  }
```

Practice

$group

```
/**
 * _id: The id of the group.
 * fieldN: The first field name.
 */
{
  _id: '$state',
  population: {
    $sum: '$pop'
  },
  zips: {
    $sum: 1
  }
}
```

Output after $group stage (Sample of 10 documents)

```
_id: "KY"
population: 3675484
zips: 809
```

```
_id: "ND"
population: 638272
zips: 391
```

$sort

```
/**
 * Provide any number of field/order pairs.
 */
{
  zips: -1
}
```

Output after $sort stage (Sample of 10 documents)

```
_id: "TX"
population: 16984601
zips: 1671
```

```
_id: "NY"
population: 17990402
zips: 1595
```

Add '$sort' stage

Practice

- $match: { *$match:* { <*query*> } }

- Q4: Show the states with more than 1000 zips

```
db.zips.aggregate ({$group:
  {
    _id: '$state',
    population: {$sum: '$pop'},
     zips: {$sum: 1}
  }
},
{$sort:
     {zips: -1}
},
{
  $match: {zips: {$gt: 1000}}
}
)
```

See result

Filter out other states

Practice

```
< {
    _id: 'TX',
    population: 16984601,
    zips: 1671
  }
  {
    _id: 'NY',
    population: 17990402,
    zips: 1595
  }
  {
    _id: 'CA',
    population: 29754890,
    zips: 1516
  }
  {
    _id: 'PA',
    population: 11881643,
    zips: 1458
  }
  {
    _id: 'IL',
    population: 11427576,
    zips: 1237
  }
  {
    _id: 'OH',
    population: 10846517,
    zips: 1007
  }
```

Practice

$group

```
/**
 * _id: The id of the group.
 * fieldN: The first field name.
 */
{
  _id: '$state',
  population: {
    $sum: '$pop'
  },
  zips: {
    $sum: 1
  }
}
```

Output after $group stage (Sample of 10 documents)

```
_id: "KY"
population: 3675484
zips: 809
```

```
_id: "ND"
population: 638272
zips: 391
```

$sort

```
/**
 * Provide any number of field/order pairs.
 */
{
  zips: -1
}
```

Output after $sort stage (Sample of 10 documents)

```
_id: "TX"
population: 16984601
zips: 1671
```

```
_id: "NY"
population: 17990402
zips: 1595
```

Add '$match' stage

$match

```
/**
 * query: The query in MQL.
 */
{
  zips: {
    $gt: 1000
  }
}
```

Output after $match stage (Sample of 6 documents)

```
_id: "TX"
population: 16984601
zips: 1671
```

```
_id: "NY"
population: 17990402
zips: 1595
```

Practice

- $match: { *$match:* { <*query*> } }

- Q1: What is the population of FISHERS ISLAND?

```
db.zips.aggregate( [
    {$match:{city: "FISHERS ISLAND"}}
])
```

See the result

- Task 1: Show only state name and location of BELL GARDENS

Practice

- $sort
  - { $sort: { <field1>: <sort order>, <field2>: <sort order> ... } }

- Q2: Sort cities of California state by population

```
db.zips.aggregate( [
    {$match:{state: "CA"}},
    {$sort:{pop: -1}}
])
```

See the result

- Task 2: Show five most populated cities of New York state (Hint: Use limit)

Practice

- $group
  - *{ $group: { _id: <expression>, <field1>: { <accumulator1> : <expression1> }, ... } }*

- Q3: Find average population of each state

```
db.zips.aggregate ({$group:
  {
    _id: '$state',
    population: {$avg: '$pop'}
  }
})
```

See the result

- Task 3: Show all states with total population of greater than 10 million and sort the result

Practice

- Q4: Show the most populated city in California state

```
db.zips.aggregate( [
  {$match:{state: "CA"}},
  {$sort:{pop: -1}},
  {$limit: 1}
])
```

See the result

- Task 4: Show the population of the most populated city in the California state using $group.

Practice

- Final Task: Create an aggregation pipeline consisting of four stages for the restaurants dataset
- Use '$group', '$match', '$sort', '$limit', '$project', etc.

- Show queries and results, and explain those

Quiz #1 (Homework)

- Download one of the datasets provided by MongoDB
  - https://github.com/neelabalan/mongodb-sample-dataset
  - https://github.com/ozlerhakan/mongodb-json-files
  - Optional: You can also use your own dataset
- Create six queries studied during the Lecture 7
  - Use $group
  - Use $unwind
  - Two, three or four stages
- Submit
  - Your dataset, your queries and their description, and result screen (report)