

Nama: Frila Cahya Wardani

Kelas: TI – 21 – PA

NPM: 212310014

Lab. Basis Data Lanjut Modul 6

1. Buatlah dan jalankan query untuk membuat setiap jenis tipe index pada database rentalfilm yang berbeda dengan contoh pada tahapan kerja.

- ensureIndex()

```
12 // Select the database to use.
13 use('rentalfilm');
14
15 // Insert a few documents into the sales collection.
16 db.customers.createIndex({age : 1})
```

| | |
|---|-------|
| 1 | age_1 |
|---|-------|

- unique

```
15 // Insert a few documents into the sales collection.
16 db.customers.createIndex({id : 1}, {unique: true})
```

| | |
|---|------|
| 1 | id_1 |
|---|------|

| | | | |
|---------|-----------|---------|---------------------------|
| > age_1 | REGULAR ⓘ | 20.5 KB | 0 (since Wed May 01 2024) |
| > id_1 | REGULAR ⓘ | 20.5 KB | 0 (since Wed May 01 2024) |

| | | | |
|--|----------|--|--|
| | UNIQUE ⓘ | | |
|--|----------|--|--|

- sparse

```
20 db.address.createIndex({"Nama_Produk" : 1}, {sparse : true})
```

| | |
|---|---------------|
| 1 | Nama_Produk_1 |
|---|---------------|

| | | | |
|-----------------|-----------|---------|---------------------------|
| > Nama_Produk_1 | REGULAR ⓘ | 20.5 KB | 0 (since Wed May 01 2024) |
|-----------------|-----------|---------|---------------------------|

| | | | |
|--|----------|--|--|
| | SPARSE ⓘ | | |
|--|----------|--|--|

- expireAfterSeconds

```
db.actor.createIndex({"last_update" : 1}, {expireAfterSeconds : 60})
```

| | |
|---|---------------|
| 1 | last_update_1 |
|---|---------------|

| | | | |
|-----------------|-----------|---------|---------------------------|
| > last_update_1 | REGULAR ⓘ | 20.5 KB | 0 (since Wed May 01 2024) |
|-----------------|-----------|---------|---------------------------|

| | | | |
|--|-------|--|--|
| | TTL ⓘ | | |
|--|-------|--|--|

- v

```
db.customers.createIndex({"email" : 1}, {v : 1})
```

| | |
|---|---------|
| 1 | email_1 |
|---|---------|

- weights

```
26 db.film.createIndex({Title : "text"}, {weights: {Title : 5}})
```

| | |
|---|------------|
| 1 | Title_text |
|---|------------|

- default_language

```
db.customers.createIndex({email : "text"}, {default_language: "english"})
```

| | |
|---|------------|
| 1 | email_text |
|---|------------|

- language_override

```
db.actor.createIndex({First_Name: "text"}, {language_override: "spanish"})
```

| | |
|---|-----------------|
| 1 | First_Name_text |
|---|-----------------|

2. Buatlah dan jalankan setiap query aggregate pada database rentalfilm yang berbeda dengan contoh yang telah diberikan.

- \$Sum

```
db.customers.aggregate([ {$group: { _id: null, age: { $sum: '$age' }}}])
```

| | |
|---|--------------|
| 1 | [|
| 2 | { |
| 3 | "_id": null, |
| 4 | "age": 24 |
| 5 | } |
| 6 |] |

- \$Avg

```
db.customers.aggregate([ {$group: { _id: null, age: { $avg: '$age' }}}])
```

| | |
|---|--------------|
| 1 | [|
| 2 | { |
| 3 | "_id": null, |
| 4 | "age": 30 |
| 5 | } |
| 6 |] |

- \$Min

```
db.customers.aggregate([ {$group: { _id: null, age: { $min: '$age' }}}])
```

| | |
|---|--------------|
| 1 | [|
| 2 | { |
| 3 | "_id": null, |
| 4 | "age": 24 |
| 5 | } |
| 6 |] |

- \$Max

```
db.customers.aggregate([ {$group: { _id: null, age: { $max: '$age' }}}])
```

| | |
|---|--------------|
| 1 | [|
| 2 | { |
| 3 | "_id": null, |
| 4 | "age": 39 |
| 5 | } |
| 6 |] |

- \$Push

```
db.actor.updateOne({id : 1}, {$push : {Usia: 40}})
```

```
1 {
2   "acknowledged": true,
3   "insertedId": null,
4   "matchedCount": 1,
5   "modifiedCount": 1,
6   "upsertedCount": 0
7 }
```

- \$addToSet

```
db.actor.updateOne({id : 1}, {$addToSet : {Usia: 70}})
```

```
1 {
2   "acknowledged": true,
3   "insertedId": null,
4   "matchedCount": 1,
5   "modifiedCount": 1,
6   "upsertedCount": 0
7 }
```

- \$First

```
db.customers.aggregate({$group: {_id: null, first: {$first: "$$ROOT"}}})
```

```
1 [
2   {
3     "_id": null,
4     "first": {
5       "_id": {
6         "$oid": "663120d3c6b740f95d7523d0"
7       },
8       "id": "Fernanda",
9       "fullname": "Fernanda Ramos",
10      "email": "fernadaramos4@uol.com.br",
11      "age": 24
12    }
13  }
14 ]
```

- \$Last

```
db.customers.aggregate({$group: {_id: null, last: {$last: "$$ROOT"}}})
```

```
1 [
2   {
3     "_id": null,
4     "last": {
5       "_id": {
6         "$oid": "6631bcca2ef4cd49b0326873"
7       },
8       "id": "Song",
9       "fullname": "Song Joong-Ki",
10      "email": "sjk@outlook.com",
11      "age": 39
12    }
13  }
14 ]
```

3. Buatlah dan jalankan setiap query pipeline pada database rentalfilm yang berbeda dengan contoh yang telah diberikan.

- \$Project

```
db.customers.aggregate([
  $project: {
    "joinedDate": 0
  }
])
```

```
1
2
3 {
4   "_id": {
5     "$oid": "663120d3c6b740f95d7523d0"
6   },
7   "id": "Fernanda",
8   "fullname": "Fernanda Ramos",
9   "email": "fernadaramos4@uo1.com.br",
10  "age": 24
11 },
12 {
13   "_id": {
14     "$oid": "663120d3c6b740f95d7523d1"
15   },
16   "id": "Mark",
17   "fullname": "Mark Daihatsu",
18   "email": "mphilips12@shaw.ca",
19   "city": "San Francisco"
20 },
21 {
22   "_id": {
23     "$oid": "663120d3c6b740f95d7523d2"
24   },
25   "id": "Jennifer",
26   "fullname": "Jennifer Aniston",
27   "email": "jenniferp@rogers.ca",
28   "occupation": "teacher"
29 },
30 {
31   "_id": {
32     "$oid": "6631bcca2ef4cd49b0326872"
33   },
34   "id": "Tom"
```

- \$Match

```
db.customers.aggregate([
  $match: {
    "fullname": "Fernanda Ramos"
  }
])
```

```
1
2
3 {
4   "_id": {
5     "$oid": "663120d3c6b740f95d7523d0"
6   },
7   "id": "Fernanda",
8   "fullname": "Fernanda Ramos",
9   "email": "fernadaramos4@uo1.com.br",
10  "age": 24
11 }
```

- \$Group

```
db.customers.aggregate([
  $group: {
    age: 27,
  }
]);
[ {age: 27}, {age: 39} ]
```

```
1
2 {
3   "age": 27
4 },
5 {
6   "age": 39
7 },
8 ]
```

- \$Sort

| | |
|---|--|
| <pre>db.customers.aggregate([\$sort: {} , { fullname: 1 }])</pre> | <pre>1 [2 { 3 "_id": { 4 "\$oid": "663120d3c6b740f95d7523d0" 5 }, 6 "id": "Fernanda", 7 "fullname": "Fernanda Ramos", 8 "email": "fernadaramos4@uol.com.br", 9 "age": 24 10 }, 11 { 12 "_id": { 13 "\$oid": "663120d3c6b740f95d7523d2" 14 }, 15 "id": "Jennifer", 16 "fullname": "Jennifer Aniston", 17 "email": "jenniferp@rogers.ca", 18 "occupation": "teacher" 19 }, 20]</pre> |
|---|--|

- \$Limit

| | |
|---|--|
| <pre>db.customers.aggregate([\$limit: 2])</pre> | <pre>1 [2 { 3 "_id": { 4 "\$oid": "663120d3c6b740f95d7523d0" 5 }, 6 "id": "Fernanda", 7 "fullname": "Fernanda Ramos", 8 "email": "fernadaramos4@uol.com.br", 9 "age": 24 10 }, 11 { 12 "_id": { 13 "\$oid": "663120d3c6b740f95d7523d1" 14 }, 15 "id": "Mark", 16 "fullname": "Mark Daihatsu", 17 "email": "mphilips12@shaw.ca", 18 "city": "San Francisco" 19 }, 20]</pre> |
|---|--|

4. Buatlah dan jalankan minimal 3 buah gabungan query pipeline pada database rentalfilm.
- Mengambil semua actor yang memiliki nama depan "Frila"

```
db.actor.aggregate([
  {
    $match: {First_Name: "Frila"}
  }
])
```

```
[
  {
    "_id": {
      "$oid": "6631213ba5f96bac26d62799"
    },
    "id": 1,
    "First_Name": "Frila",
    "Last_Name": "Cw",
    "Last_Update": "2024-03-11 10:46:54",
    "Usia": [
      40,
      70
    ]
  },
  {
    "_id": {
      "$oid": "6631213ba5f96bac26d6279a"
    },
    "id": 2,
    "First_Name": "Frila",
    "Last_Name": "Cahya",
    "Last_Update": "2024-03-11 10:46:54"
  },
  {
    "_id": {
      "$oid": "6631213ba5f96bac26d6279b"
    },
    "id": 3,
    "First_Name": "Frila",
    "Last_Name": "Wardani",
    "Last_Update": "2024-03-11 10:46:54"
  }
]
```

- Mengambil dua film dengan rating tertinggi

```
db.film.aggregate([
  {
    $sort: {
      Rating: -1
    }
  },
  {
    $limit: 2
  }
])
```

```
[
  {
    "_id": {
      "$oid": "6631c6b27f35fb7ab70e6f0b"
    },
    "id": 2,
    "Title": "Exhuma",
    "Release_Year": 2024,
    "Genre": "Horror",
    "Director": "Jang Jae-Hyun",
    "Rating": 5
  },
  {
    "_id": {
      "$oid": "6631c6757f35fb7ab70e6f0a"
    },
    "id": 3,
    "Title": "Mission Impossible: Fallout",
    "Release_Year": 2018,
    "Genre": "Action",
    "Director": "Christopher",
    "Rating": 5
  }
]
```

- Menghitung jumlah film yang dirilis pada setiap tahun

| | |
|---|--|
| <pre> db.film.aggregate([\$group: { _id: "\$Release_Year", count: { \$sum: 1 } }]) </pre> | <pre> 1 [2 { 3 "_id": 2023, 4 "count": 2 5 }, 6 { 7 "_id": 2018, 8 "count": 1 9 }, 10 { 11 "_id": 2024, 12 "count": 1 13 } 14] </pre> |
|---|--|

- Menggabungkan data actor dan film yang mereka bintang

| | |
|--|---|
| <pre> db.actor.aggregate([\$lookup: { from: "actor_film", localField: "Id", foreignField: "Actor Id", as: "films" }]) </pre> | <pre> 60 { 61 "_id": { 62 "\$oid": "6631c8aabff744428b27dff2" 63 }, 64 "id": 6, 65 "Fisrt_Name": "Song", 66 "Last_Name": "Joong-Ki", 67 "Last_Update": { 68 "\$date": "2024-05-01T04:44:26.409Z" 69 }, 70 "films": [] 71 }, 72 { 73 "_id": { 74 "\$oid": "6631c8aabff744428b27dff3" 75 }, 76 "id": 7, 77 "Fisrt_Name": "Tom", 78 "Last_Name": "Cruise", 79 "Last_Update": { 80 "\$date": "2024-05-01T04:44:26.409Z" 81 }, 82 "films": [] 83 } 84] </pre> |
|--|---|

- Menghitung rata-rata durasi film

| | |
|--|---|
| <pre> db.film.aggregate([\$group: { _id: null, averageLength: { \$avg: "\$Duration" } }]) </pre> | <pre> 1 [2 { 3 "_id": null, 4 "averageLength": 136.75 5 } 6] </pre> |
|--|---|