

# Bazy danych – NoSQL

## MongoDB – zadania

Rafał Gawęł

Laboratorium – 21.11.2019r.

Wykonano – 01.12.2019r.

1. Wykorzystując bazę danych **yelpdataset** wykonaj zapytanie i komendy MongoDB, aby uzyskać następujące rezultaty:
  - a. Zwróć bez powtórzeń wszystkie nazwy miast w których znajdują się firmy(business):

```
db.business.distinct("city")
```

Rezultat:

Key	Value	Type
▼ (1)	[ 172 elements ]	Array
[0]	Phoenix	String
[1]	Mc Farland	String
[2]	De Forest	String
[3]	Middleton	String
[4]	Madison	String
[5]	Sun Prairie	String
[6]	Windsor	String
[7]	Monona	String
[8]	Chandler	String
[9]	Scottsdale	String
[10]	Tempe	String
[11]	Florence	String
[12]	Peoria	String
[13]	Glendale	String
[14]	Cave Creek	String
[15]	Paradise Valley	String
[16]	Mesa	String
[17]	Ahwatukee	String
[18]	Phoenix	String
[19]	Anthem	String
[20]	Gilbert	String
[21]	Gold Canyon	String
[22]	Apache Junction	String
[23]	Goldfield	String
[24]	Casa Grande	String
[25]	Coolidge	String
[26]	Queen Creek	String
[27]	Higley	String
[28]	Sun Lakes	String

- b. Zwróć liczbę wszystkich recenzji, które pojawiły się w roku 2011 i 2012.

```
db.review.count (
  {date:
    {$regex: /^(2011)|2012/}
  })
```

Rezultat:

2.09 sec.  
382376

- c. Zwróć dane wszystkich otwartych(open) firm(business) z pól: id, nazwa, adres

```
db.business.find(
  {open:true},
  {business_id:1, full_address:1, name:1, _id:0}
)
```

Rezultat:

business 0.039 sec.		
Key	Value	Type
full_address	1801 Deming Way Middleton, WI 53562	String
name	Hilton Garden Inn Madison West/Middleton	String
(45)	{ 3 fields }	Object
business_id	yulcDVSy03FQe1RWWhicDQw	String
full_address	2540 Allen Blvd Middleton, WI 53562	String
name	Ace Hardware	String
(46)	{ 3 fields }	Object
business_id	elBgDynz6UPoSqQgixggJw	String
full_address	3201 Glacier Ridge Rd Middleton, WI 53562	String
name	Revolution Hair Company Inc	String
(47)	{ 3 fields }	Object
business_id	yNayFJemmt-J3fNmWuKA	String
full_address	7425 Hubbard Ave Middleton, WI 53562	String
name	Middleton Public Pibrary	String
(48)	{ 3 fields }	Object
business_id	_tnYCSHxuDIXjNVNctTkCA	String
full_address	6323 University Ave Middleton, WI 53562	String
name	Dick & Arnie's Barber Shop	String
(49)	{ 3 fields }	Object
business_id	naqKXvhoqljvA7kuo15gEw	String
full_address	8212 Greenway Blvd Middleton, WI 53562	String
name	Fairfield Inn & Suites Madison West	String
(50)	{ 3 fields }	Object
business_id	IN-5-YTsaJr_lByyA476iw	String
full_address	6815 University Ave Middleton, WI 53562	String
name	Middleton Sport Bowl	String

- d. Zwróć dane wszystkich użytkowników(user), którzy uzyskali przynajmniej jeden pozytywny głos z jednej kategorii (funny, useful, cool), wynik posortuj alfabetycznie na podstawie imienia użytkownika.

```
db.user.createIndex({name:1})
```

Key	Value	Type
(1)	{ 4 fields }	Object
numIndexesBefore	2	Int32
numIndexesAfter	2	Int32
note	all indexes already exist	String
ok	1.0	Double

```

db.user.find({
  $or:[
    {'votes.funny': {$gt:0}},
    {'votes.useful': {$gt:0}},
    {'votes.cool': {$gt:0}}
  ]
}).sort({name:1})

```

## Rezultat:

user 0.006 sec.		
Key	Value	Type
> votes	{ 3 fields }	Object
review_count	10	Int32
name	Cody	String
user_id	fCNbLqIPsQfA/9Taj12Q	String
> friends	[ 1 element ]	Array
fans	1	Int32
average_stars	3.56	Double
type	user	String
> compliments	{ 0 fields }	Object
> elite	[ 0 elements ]	Array
> (2) ObjectId("5de3c91001f8f52cc8ffd77e")	{ 12 fields }	Object
> (3) ObjectId("5de3c91001f8f52cc8ffd7e")	{ 12 fields }	Object
> (4) ObjectId("5de3c90b01f8f52cc8feab6e")	{ 12 fields }	Object
> (5) ObjectId("5de3c91401f8f52cc800d542")	{ 12 fields }	Object
> (6) ObjectId("5de3c91701f8f52cc8018d5b")	{ 12 fields }	Object
> (7) ObjectId("5de3c91001f8f52cc8fff7fe")	{ 12 fields }	Object
> (8) ObjectId("5de3c91701f8f52cc801b9e2")	{ 12 fields }	Object
> (9) ObjectId("5de3c90f01f8f52cc8ff9803")	{ 12 fields }	Object
> (10) ObjectId("5de3c91001f8f52cc8ffe1d1")	{ 12 fields }	Object
> (11) ObjectId("5de3c90e01f8f52cc8ff8dd0")	{ 12 fields }	Object
> (12) ObjectId("5de3c91101f8f52cc80010f3")	{ 12 fields }	Object
> (13) ObjectId("5de3c91601f8f52cc80178fa")	{ 12 fields }	Object
> (14) ObjectId("5de3c90901f8f52cc8fe2785")	{ 12 fields }	Object
> (15) ObjectId("5de3c91001f8f52cc8ff8a1")	{ 12 fields }	Object

- e. Określ, ile każde przedsiębiorstwo otrzymało napiwków w 2013. Wynik posortuj alfabetycznie na podstawie nazwy firmy.

```
db.tip.aggregate([
  {$match: {date: {$regex: /^2013/}}},
  {
    $lookup:{
      "from": "business",
      "localField": "business_id",
      "foreignField": "business_id",
      "as": "business"
    }
  },
  {
    $unwind:"$business"
  },
  {
    $group:
      {
        _id:"$business.name",
        count: {$sum:1}
      }
  },
  {
    $project:{
      _id:0,
      name:"$_id",
      count:1
    }
  },
  {
    $sort:{
      name:1
    }
  }
])
```

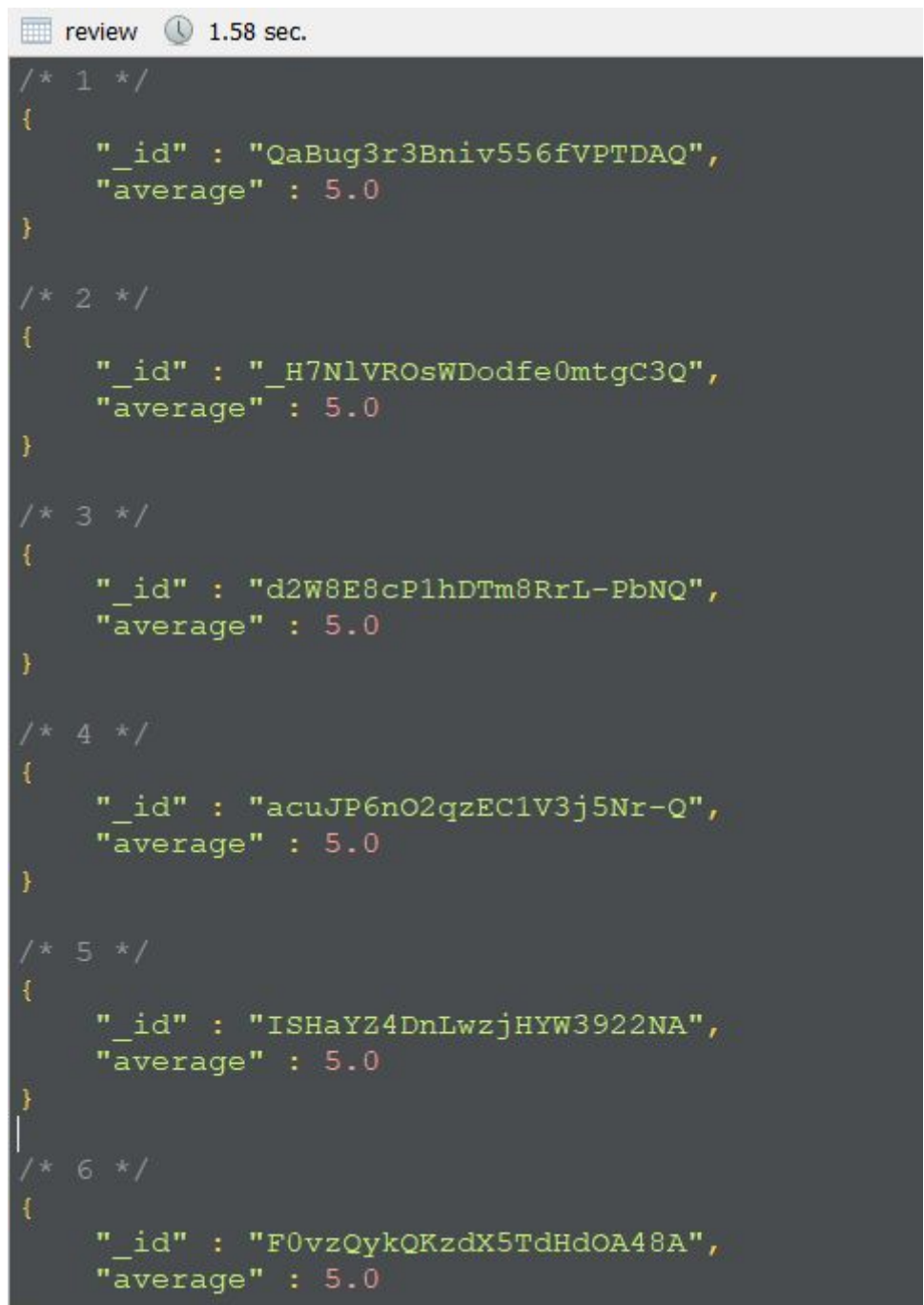
Powyższa funkcja działa jednak bardzo długo. Żeby doczekać końca można nałożyć limit.

- f. Wyznacz, jaką średnią ocen (stars) uzyskała każda firma (business) na podstawie wszystkich recenzji, wynik posortuj od najwyższego uzyskanego wyniku.

Wersja prosta(nie pokazuje nazwy firmy):

```
db.review.aggregate([
  {$group: { _id: "$business_id", average: { $avg: "$stars" } }},
  {$sort: {average: -1}}
])
```

Rezultat:



```
review 1.58 sec.
/* 1 */
{
  "_id" : "QaBug3r3Bniv556fVPTDAQ",
  "average" : 5.0
}

/* 2 */
{
  "_id" : "_H7NlVROsWDodfe0mtgC3Q",
  "average" : 5.0
}

/* 3 */
{
  "_id" : "d2W8E8cPlhDTm8RrL-PbNQ",
  "average" : 5.0
}

/* 4 */
{
  "_id" : "acuJP6nO2qzEC1V3j5Nr-Q",
  "average" : 5.0
}

/* 5 */
{
  "_id" : "ISHaYZ4DnLwzjHYW3922NA",
  "average" : 5.0
}

/* 6 */
{
  "_id" : "F0vzQykQKzdX5TdHdOA48A",
  "average" : 5.0
}
```

Wersja w której widać nazwę firmy, ale działa bardzo wolno:

```
db.review.aggregate([
  {$group: { _id: "$business_id", average: { $avg: "$stars" } }},
  {$sort: {average: -1}}
]).map(function(x) {
  return {
    "name": db.business.findOne({business_id: x._id}).name,
    "tips": x.average
  };
});
```

g. Usuń wszystkie firmy, które mają ocenę (stars) poniżej 3.

```
db.business.remove({stars:{$lt:3}})
```

Rezultat:

```
🕒 0.322 sec.
Removed 6339 record(s) in 322ms
```

2. Zdefiniuj funkcję (MongoDB) umożliwiającą dodanie nowej wskazówki/napiwku (tip). Wykonaj przykładowe wywołanie

```
function newTip(user_id, text, business_id, description, likes, date) {
  var argsOk = true;

  argsOk &= business !== null && business.business_id !== null;
  argsOk &= user !== null && user.user_id !== null;

  if(!argsOk){
    return null;
  }

  var tip = {
    user_id: user.user_id,
    text: text,
    business_id: business.business_id,
    likes: likes,
    date: { date:ISODate(date) },
    type: "tip",
    _id: new ObjectId().valueOf()
  };

  var result = db.tip.insert(tip);
  return (result.nInserted === 1) ? tip : null;
}
```

Przykładowe wywołanie:

```
var user = db.user.findOne();
var business = db.business.findOne();
newTip(user, 'Delicious', business, 0, '2019-12-01')
```

Rezultat:

```
{
  "_id" : "5de6bb0b059c101104f6edfe",
  "user_id" : "0vscrHoajVRa1Yk19XWdwA",
  "text" : "Delicious",
  "business_id" : "vcNAWiLM4dR7D2nwwJ7nCA",
  "likes" : "2019-12-01",
  "date" : {
    "date" : ISODate("2019-12-03T19:44:11.879Z")
  },
  "type" : "tip"
}
```

3. Zdefiniuj funkcję (*MongoDB*), która zwróci wszystkie wskazówki/napiwki (*tip*), w których w tekście znajdzie się fraza podana jako argument. Wykonaj przykładowe wywołanie zdefiniowanej funkcji.

```
function byTip(substring) {
  return db.tip.find({text: {$regex: "/" + substring + "/"}});
}
```

```
byTip('Garden');
```

Rezultat

```
{
  "_id" : ObjectId("5de3c7bf1a2270292419elec"),
  "user_id" : "karWoeb37ggm5_4V9-jg2A",
  "text" : "Lumber/Building Materials and Garden Supplies/Paint are on opposite sides of the store.",
  "business_id" : "-52nir13iFnSMdBew674HA",
  "likes" : 0,
  "date" : "2013-07-08",
  "type" : "tip"
}
```



4. Zdefiniuj funkcję (*MongoDB*), która umożliwi modyfikację nazwy firmy (*business*) na podstawie id. Id oraz nazwa mają być przekazywane jako parametry.

```
function updateName(id, newName) {
  var business = db.business.findOne({"business_id": id});
  if(business === null) {
    return null;
  }
  business.name = newName;
  var result = db.business.save(business);
  return (result.nMatched === 1) ? business : null;
}
```

Wywołanie:

```
updateName("vcNAWiLM4dR7D2nwwJ7nCA", "Firemka");
```

Przed:

```
{
  "Thursday" : {
    "close" : "17:00",
    "open" : "08:00"
  },
  "open" : true,
  "categories" : [
    "Doctors",
    "Health & Medical"
  ],
  "city" : "Phoenix",
  "review_count" : 7,
  "name" : "Eric Goldberg, MD",
  "neighborhoods" : [],
  "longitude" : -111.983758,
  "state" : "AZ",
  "stars" : 3.5,
  "latitude" : 33.499313,
  "attributes" : {
    "By Appointment Only" : true
  },
  "type" : "business"
}
```



Po:

```
    },
    "Thursday" : {
      "close" : "17:00",
      "open" : "08:00"
    }
  },
  "open" : true,
  "categories" : [
    "Doctors",
    "Health & Medical"
  ],
  "city" : "Phoenix",
  "review_count" : 7,
  "name" : "Firemka",
  "neighborhoods" : [],
  "longitude" : -111.983758,
  "state" : "AZ",
  "stars" : 3.5,
  "latitude" : 33.499313,
  "attributes" : {
    "By Appointment Only" : true
  },
  "type" : "business"
}
```

5. Zwróć średnią ilość wszystkich recenzji użytkowników, wykorzystaj map reduce.

```
db.user.mapReduce(
  function(){ emit(1, this.review_count) },
  function(name, reviews_count){
    return Array.avg(reviews_count)
  },
  {out:"out"}
);
```

Rezultat:

```
out 0.001 sec.
/* 1 */
{
  "_id" : 1.0,
  "value" : 33.3711549796441
}
```

6. Odwzoruj wszystkie zadania z punktu 1 w języku programowania (np. JAVA) z pomocą API do MongoDB. Wykorzystaj dla każdego zadania odrębną metodę.
- a. Zwróć bez powtórzeń wszystkie nazwy miast w których znajdują się firmy (*business*).

```
private void showDistinctBusinessCities(){
    DBCollection coll = db.getCollection( name: "business");
    BasicDBObject query = new BasicDBObject();
    List list = coll.distinct( fieldName: "city");
    for (Object tmp : list) System.out.println(tmp);
}
```

```
INFO: Opened connection [connectionId{localValue:2, serverValue:65}] to 127.0.0.1:27017
Phoenix
De Forest
Mc Farland
Middleton
Madison
Sun Prairie
Windsor
Monona
```

- b. Zwróć liczbę wszystkich recenzji, które pojawiły się w roku 2011 i 2012.

```
private void countReviewsIn2011And2012(){
    DBCollection coll = db.getCollection( name: "review");
    BasicDBObject query = new BasicDBObject();
    Pattern pattern = Pattern.compile("(2011).*|(2012).*");
    query.append( key: "date", pattern);
    DBCursor cursor = coll.find(query);
    System.out.println(cursor.length());
}
```

Rezultat:

```
INFO: Opened co
382376
```

- c. Zwróć dane wszystkich otwartych(open) firm(business) z pól: id, nazwa, adres

```
private void findOpenBusiness(){
    DBCollection coll = db.getCollection( name: "business");
    BasicDBObject query = new BasicDBObject("open", true);
    BasicDBObject fields = new BasicDBObject("business_id",true)
        .append( key: "name", val: true)
        .append( key: "full_address", val: true)
        .append( key: "_id", val: false);
    DBCursor cursor = coll.find(query, fields);
    while (cursor.hasNext()) {
        System.out.println(cursor.next());
    }
    cursor.close();
}
```

Rezultat:

```
ender Sun, 11 03:22 : name = find instances /
{"business_id": "FnJNNUQRwB1_AdF48tosQ", "full_address": "3385 S Durango\nSpring Valley\nLas Vegas, NV 89117", "name": "aDias Salon"}
{"business_id": "B6uTC4FeLRCKETNs-HBPTg", "full_address": "2321 N Rainbow Blvd\nLas Vegas, NV 89108", "name": "Freaking Bros The Trilogy of Terror"}
{"business_id": "Gdmig8uYU5Mdfcg0XsT3Q", "full_address": "4730 E Indian School Rd\nSte 120\nPhoenix, AZ 85018", "name": "Black Cat Coffee House"}
{"business_id": "bSN-79IH8z2JefG0knAUUQ", "full_address": "950 E Pecos Rd\nChandler, AZ 85225", "name": "Pretty Pretty Please"}
{"business_id": "h4y4A11AUQrZovRipOvjwA", "full_address": "6149 S Rainbow Blvd\nSpring Valley\nLas Vegas, NV 89118", "name": "eRealty"}
{"business_id": "tWVuAPWxvZg8ce0WepXA", "full_address": "20206 N 27th Ave\nSte A\nPhoenix, AZ 85027", "name": "George's Famous Gyros & Pasta"}
{"business_id": "0dLSUMd7ds3fLU0Iz6RRA", "full_address": "9802 N 59th Ave\nGlendale, AZ 85302", "name": "Saguaro Ranch Park"}
{"business_id": "-_ghWASvFpxiPuhzRb8kw", "full_address": "3545 S Fort Apache Rd\nSte 135\nSpring Valley\nLas Vegas, NV 89147", "name": "Vegas Cellular Solutions"}
{"business_id": "GuT460v9AlHHIqArL32i9Q", "full_address": "11460 W Hilton Way\nAvondale, AZ 85323", "name": "Garden Grille & Bar"}
{"business_id": "s4gSQGzm9G75JZd3RRVWFA", "full_address": "8046 North 19th Ave\nPhoenix, AZ 85021", "name": "The Studio Academy of Beauty"}
{"business_id": "aAM72d011G0GTE32uGbu", "full_address": "5000 N 19th Ave\nPhoenix, AZ 85016", "name": "Downtown Southville Event(*)"}
```

- d. Zwróć dane wszystkich użytkowników(user), którzy uzyskali przynajmniej jeden pozytywny głos z jednej kategorii (funny, useful, cool), wynik posortuj alfabetycznie na podstawie imienia użytkownika

```
private void findUsers(){
    DBCollection coll = db.getCollection( name: "user");
    DBObject clause1 = new BasicDBObject("votes.funny", new BasicDBObject("$gt", 0));
    DBObject clause2 = new BasicDBObject("votes.useful", new BasicDBObject("$gt", 0));
    DBObject clause3 = new BasicDBObject("votes.cool", new BasicDBObject("$gt", 0));
    BasicDBList or = new BasicDBList();
    or.add(clause1);
    or.add(clause2);
    or.add(clause3);
    DBObject query = new BasicDBObject("$or", or);
    DBCursor cursor = coll.find(query).sort(new BasicDBObject("name",1));
    while (cursor.hasNext()) {
        System.out.println(cursor.next());
    }
    cursor.close();
}
```

```
INFO: Opened connection [connectionid(localhost:2, servervalue:151)] to 127.0.0.1:27017
{"_id": {"$oid": "5de3c98e01f8f52cc8ff7f8c"}, "yelping_since": "2009-08", "votes": {"funny": 2, "useful": 7, "cool": 0}, "review_count": 10, "name": "Cody", "user_id": "fCnBilqir"},
{"_id": {"$oid": "5de3c91001f8f52cc8ffdd7e"}, "yelping_since": "2009-07", "votes": {"funny": 10, "useful": 39, "cool": 7}, "review_count": 17, "name": "Kurt", "user_id": "meqaap"},
{"_id": {"$oid": "5de3c91001f8f52cc8ffdd7e"}, "yelping_since": "2011-12", "votes": {"funny": 0, "useful": 2, "cool": 2}, "review_count": 10, "name": "Anastacia", "user_id": "qJL"},
{"_id": {"$oid": "5de3c90b01f8f52cc8feab6e"}, "yelping_since": "2012-03", "votes": {"funny": 0, "useful": 1, "cool": 1}, "review_count": 2, "name": "Brandon", "user_id": "CfWwBC"},
{"_id": {"$oid": "5de3c91401f8f52cc800d542"}, "yelping_since": "2011-10", "votes": {"funny": 0, "useful": 3, "cool": 1}, "review_count": 5, "name": "David", "user_id": "M5IQ6r6G"},
{"_id": {"$oid": "5de3c91701f8f52cc8018d5b"}, "yelping_since": "2011-04", "votes": {"funny": 0, "useful": 5, "cool": 3}, "review_count": 6, "name": "Jenelle", "user_id": "LI5yKf"},
{"_id": {"$oid": "5de3c91001f8f52cc8ff7fe"}, "yelping_since": "2011-05", "votes": {"funny": 3, "useful": 9, "cool": 6}, "review_count": 18, "name": "Nancy", "user_id": "6n82GjK"},
{"_id": {"$oid": "5de3c91701f8f52cc801b9e2"}, "yelping_since": "2008-04", "votes": {"funny": 5, "useful": 11, "cool": 4}, "review_count": 19, "name": "Nette", "user_id": "piwME0"},
{"_id": {"$oid": "5de3c90f01f8f52cc8ff9803"}, "yelping_since": "2013-11", "votes": {"funny": 0, "useful": 1, "cool": 2}, "review_count": 4, "name": "Persian", "user_id": "bSoPYS"}
```

- e. Określ, ile każde przedsiębiorstwo otrzymało napiwków w 2013. Wynik posortuj alfabetycznie na podstawie nazwy.

```
private void findTipsIn2013ByBusiness(){
    DBCollection coll = db.getCollection( name: "tip");

    BasicDBObject query = new BasicDBObject();
    Pattern pattern = Pattern.compile("(2013).*");
    query.append( key: "date", pattern);

    Iterable<DBObject> output = coll.aggregate(Arrays.asList(
        new BasicDBObject("$limit", 1000),
        new BasicDBObject("$match", query),
        new BasicDBObject("$lookup", new BasicDBObject("from", "business")
            .append( key: "localField", val: "business_id")
            .append( key: "foreignField", val: "business_id")
            .append( key: "as", val: "business")),
        new BasicDBObject("$unwind", "$business"),
        new BasicDBObject("$group", new BasicDBObject("_id", "$business.name")
            .append( key: "count", new BasicDBObject("$sum", 1))),
        new BasicDBObject("$project", new BasicDBObject("_id", 0)
            .append( key: "name", val: "$_id")
            .append( key: "count", val: 1)),
        new BasicDBObject("$sort", new BasicDBObject("name", 1))
    )).results();

    for (DBObject dbObject : output)
    {
        System.out.println(dbObject);
    }
}
```



INFO: Opened connection [connectionId{localValue:2, serverValue:215}] to 127.0

```
{"count": 1, "name": "007 Nails"}
{"count": 1, "name": "Angelo's"}
{"count": 1, "name": "Arizona School For the Arts"}
{"count": 2, "name": "Barriques Wine & Spirits"}
{"count": 2, "name": "Barrymore Theatre"}
{"count": 2, "name": "Beach House Restaurant & Lounge"}
{"count": 1, "name": "Benvenuto's Italian Grill"}
{"count": 1, "name": "Best Western Inntowner And The Highland Club"}
{"count": 1, "name": "Bowling Green Rec Center"}
{"count": 2, "name": "Cafe Zoma"}
{"count": 13, "name": "Canyon Cafe"}
{"count": 2, "name": "Capital Brewery & Beer Garden"}
{"count": 1, "name": "Central Park Square Athletic Club"}
{"count": 3, "name": "Chico's Tacos"}
{"count": 4, "name": "Chin's Asia Fresh"}
{"count": 12, "name": "Comedor Guadalajara"}
{"count": 2, "name": "Cool Beans Coffee House and Cafe"}
{"count": 1, "name": "Copps Food Center"}
{"count": 1, "name": "Copps Foods"}
{"count": 2, "name": "Cottage Cafe"}
{"count": 1, "name": "Country Cafe"}
{"count": 1, "name": "Courtyard by Marriot - Madison East"}
{"count": 2, "name": "Culver's"}
{"count": 1, "name": "David's Jamaican Cuisine"}
{"count": 1, "name": "DeChance & Company"}
{"count": 2, "name": "Deforest Family Restaurant"}
```



- f. Wyznacz, jaką średnią ocen (stars) uzyskała każda firma (business) na podstawie wszystkich recenzji, wynik posortuj od najwyższego uzyskanego wyniku.

```
private void avgStars(){
    DBCollection coll = db.getCollection( name: "review");

    Iterable<DBObject> output = coll.aggregate(Arrays.asList(
        new BasicDBObject("$limit", 1000),
        new BasicDBObject("$group", new BasicDBObject("_id", "$business_id")
            .append( key: "avgStars", new BasicDBObject("$avg", "$stars"))),
        new BasicDBObject("$lookup", new BasicDBObject("from", "business")
            .append( key: "localField", val: "_id")
            .append( key: "foreignField", val: "business_id")
            .append( key: "as", val: "business")),
        new BasicDBObject("$unwind", "$business"),
        new BasicDBObject("$project", new BasicDBObject("avgStars", 1)
            .append( key: "_id", val: 0)
            .append( key: "name", val: "$business.name")),
        new BasicDBObject("$sort", new BasicDBObject("avgStars", -1))
    )).results();

    for (DBObject dbObject : output)
    {
        System.out.println(dbObject);
    }
}
```

```
INFO: Opened connection [connectionId{localValue:2, serverValue:239}] to 127.0.0.1:27017
{"avgStars": 5.0, "name": "Bennett's Auto Repair LLC"}
{"avgStars": 5.0, "name": "All Pets Veterinary Clinic"}
{"avgStars": 5.0, "name": "Revolution Hair Company Inc"}
{"avgStars": 5.0, "name": "Dunn's Import Inc"}
{"avgStars": 5.0, "name": "Ace Hardware"}
{"avgStars": 5.0, "name": "Killian Dental Clinic"}
{"avgStars": 5.0, "name": "Brandon Eyes"}
{"avgStars": 5.0, "name": "Lori's Pet-Agree Salon Llc"}
{"avgStars": 5.0, "name": "Sun Prairie Public Library"}
{"avgStars": 5.0, "name": "Prairie Land Service Center"}
{"avgStars": 5.0, "name": "Magic Wash Inc"}
{"avgStars": 4.8, "name": "Harbor Athletic Club"}
{"avgStars": 4.75, "name": "Parsonage Bed & Breakfast the"}
{"avgStars": 4.714285714285714, "name": "Staybridge Suites Extended Stay Hotel Middleton/Madison-West"}
```

- g. Usuń wszystkie firmy, które mają ocenę (stars) poniżej 3.

```
private void deleteBusinessWithLT3Stars(){
    DBCollection coll = db.getCollection( name: "business");
    BasicDBObject query = new BasicDBObject("stars", new BasicDBObject("$lt", 3));
    coll.remove(query);
}
```

7. Zaproponuj bazę danych składającą się z 3 kolekcji pozwalającą przechowywać dane dotyczące: studentów, przedmiotów oraz sal zajęciowych. W bazie wykorzystaj: pola proste, złożone i tablice. Zaprezentuj strukturę dokumentów w formie JSON dla przykładowych danych.

```
db.class.insertMany([
  {
    _id: "1.38",
    building: "D17",
    floor: 1,
  },
  {
    _id: "2.41",
    building: "D17",
    floor: 2,
  },
  {
    _id: "4.28",
    building: "D17",
    floor: 4,
  }
])
```

```
db.subject.insertMany([
  {
    _id: "Bazy Danych",
    ECTS: 3,
    tutor: "Jan Kowalski",
    class: {$ref: "class", $id: "4.28"}
  },
  {
    _id: "WDI",
    ECTS: 6,
    tutor: "Jan Nowak",
    class: {$ref: "class", $id: "2.41"}
  },
  {
    _id: "ASD",
    ECTS: 6,
    tutor: "Nowak Kowalski",
    class: {$ref: "class", $id: "1.38"}
  }
])
```

```

db.student.insertMany([
{
  name: "Rafał",
  surname: "Gaweł",
  year: 3,
  subjects:[
    {subject: {$ref: "subject", $id: "Bazy Danych"}},
    {subject: {$ref: "subject", $id: "ASD"}}
  ]
},
{
  name: "Marcin",
  surname: "Gaweł",
  year: 1,
  subjects:[
    {subject: {$ref: "subject", $id: "Bazy Danych"}},
    {subject: {$ref: "subject", $id: "WDI"}}
  ]
}
])

```

Rezultat:

```

db.getCollection('class').find({})

```

class 0.001 sec.

```

/* 1 */
{
  "_id" : "1.38",
  "building" : "D17",
  "floor" : 1.0
}

/* 2 */
{
  "_id" : "2.41",
  "building" : "D17",
  "floor" : 2.0
}

/* 3 */
{
  "_id" : "4.28",
  "building" : "D17",
  "floor" : 4.0
}

```

```
/* 1 */
{
  "_id" : "Bazy Danych",
  "ECTS" : 3.0,
  "tutor" : "Jan Kowalski",
  "class" : {
    "$ref" : "class",
    "$id" : "4.28"
  }
}

/* 2 */
{
  "_id" : "WDI",
  "ECTS" : 6.0,
  "tutor" : "Jan Nowak",
  "class" : {
    "$ref" : "class",
    "$id" : "2.41"
  }
}

/* 3 */
{
  "_id" : "ASD",
  "ECTS" : 6.0,
  "tutor" : "Nowak Kowalski",
  "class" : {
    "$ref" : "class",
    "$id" : "1.38"
  }
}
}
```

```

/* 1 */
{
  "_id" : ObjectId("5dee7f851d2bbecf22383121"),
  "name" : "Rafał",
  "surname" : "Gaweł",
  "year" : 3.0,
  "subjects" : [
    {
      "subject" : {
        "$ref" : "subject",
        "$id" : "Bazy Danych"
      }
    },
    {
      "subject" : {
        "$ref" : "subject",
        "$id" : "ASD"
      }
    }
  ]
}

```

```

/* 2 */
{
  "_id" : ObjectId("5dee7f851d2bbecf22383122"),
  "name" : "Marcin",
  "surname" : "Gaweł",
  "year" : 1.0,
  "subjects" : [
    {
      "subject" : {
        "$ref" : "subject",
        "$id" : "Bazy Danych"
      }
    },
    {
      "subject" : {
        "$ref" : "subject",
        "$id" : "WDI"
      }
    }
  ]
}

```