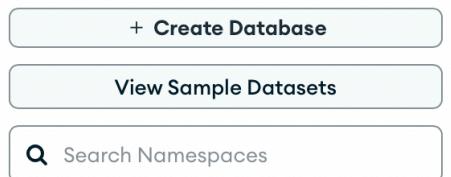


**Julia Smerdel**  
**MongoDB**  
**Zadanie domowe**

1a)



```
▶ JS
└── businessDB
    └── business
```

```
_id: ObjectId('6477b04bd7bc13d115a46835')
business_... : "ZW2WeP2Hp20tq0RG1NFkoQ"
full_address: "4709 Triangle St
               Mc Farland, WI 53558"
▶ hours: Object
  op... : true
▶ categories: Array
  ci... : "Mc Farland"
  review_cou... : 4
  na... : "Air Quality Systems"
▶ neighborhoo... : Array
  ...
```

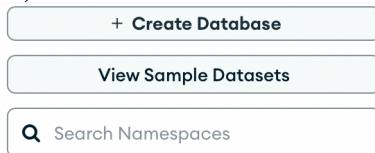
```
db.business.find(
{
  categories: "Restaurants",
  "hours.Monday": {$exists: true},
  stars: {$gte: 4}
},
{
  name: true,
  full_address: true,
  categories: true,
  hours: true,
  stars: true,
}
).sort({name: 1});
```

```
{
  _id: ObjectId("6477b080d7bc13d115a4c615"),
  full_address: '7051 E 5th Ave\nScottsdale, AZ 85251',
  hours: {
    Monday: { close: '21:00', open: '11:00' },
    Tuesday: { close: '22:00', open: '11:00' },
    Friday: { close: '23:00', open: '11:00' },
    Wednesday: { close: '22:00', open: '11:00' },
    Thursday: { close: '22:00', open: '11:00' },
    Sunday: { close: '21:00', open: '11:00' },
    Saturday: { close: '23:00', open: '11:00' }
  },
  categories: [
    'Wine Bars',
    'Bars',
    'American (New)',
    'Nightlife',
    'Restaurants'
  ],
  name: '5th and Wine',
  stars: 4
},
{
  _id: ObjectId("6477b0b0d7bc13d115a50cd1"),
  full_address: '6475 W Charleston Blvd\nSte 150\nWestside\nLas Vegas, NV 89146',
  hours: {
    Monday: { close: '20:00', open: '11:00' },
    Tuesday: { close: '20:00', open: '11:00' },
    Friday: { close: '20:00', open: '11:00' },
    Wednesday: { close: '20:00', open: '11:00' },
    Thursday: { close: '20:00', open: '11:00' },
    Sunday: { close: '20:00', open: '11:00' },
    Saturday: { close: '20:00', open: '11:00' }
  },
  categories: [ 'Chicken Wings', 'Restaurants' ],
  name: '702 Wing Spot',
  stars: 4.5
}
]
```

```
db.business.aggregate(  
{  
    $match: {  
        $or: [  
            {categories: "Hotels & Travel"},  
            {categories: "Hotels"}  
        ]  
    }  
},  
{  
    $group:{  
        _id: "$city",  
        hotelsCounter: {$sum:1}  
    }  
},  
{  
    $sort: {hotelsCounter: 1}  
}  
)
```

```
[Atlas atlas-84ca13-shard-0 [primary] businessDB> it
[
  { _id: 'Maricopa', hotelsCounter: 1 },
  { _id: 'Buckeye', hotelsCounter: 1 },
  { _id: 'Fort McDowell', hotelsCounter: 1 },
  { _id: 'Ratho', hotelsCounter: 1 },
  { _id: 'Youngtown', hotelsCounter: 1 },
  { _id: 'Heidelberg', hotelsCounter: 1 },
  { _id: 'Carefree', hotelsCounter: 2 },
  { _id: 'Fitchburg', hotelsCounter: 2 },
  { _id: 'Anthem', hotelsCounter: 2 },
  { _id: 'Clark County', hotelsCounter: 2 },
  { _id: 'South Queensferry', hotelsCounter: 2 },
  { _id: 'Cambridge', hotelsCounter: 2 },
  { _id: 'Gold Canyon', hotelsCounter: 2 },
  { _id: 'Litchfield Park', hotelsCounter: 2 },
  { _id: 'Queen Creek', hotelsCounter: 2 },
  { _id: 'Lasswade', hotelsCounter: 2 },
  { _id: 'Florence', hotelsCounter: 3 },
  { _id: 'Cave Creek', hotelsCounter: 3 },
  { _id: 'Apache Junction', hotelsCounter: 3 },
  { _id: 'Paradise Valley', hotelsCounter: 3 }
]
Type "it" for more
[Atlas atlas-84ca13-shard-0 [primary] businessDB> it
[
  { _id: 'Tolleson', hotelsCounter: 3 },
  { _id: 'Fountain Hills', hotelsCounter: 4 },
  { _id: 'Gilbert', hotelsCounter: 4 },
  { _id: 'Avondale', hotelsCounter: 5 },
  { _id: 'Kitchener', hotelsCounter: 5 },
  { _id: 'Waterloo', hotelsCounter: 5 },
  { _id: 'Wickenburg', hotelsCounter: 5 },
  { _id: 'Gila Bend', hotelsCounter: 6 },
  { _id: 'Middleton', hotelsCounter: 7 },
  { _id: 'Casa Grande', hotelsCounter: 7 },
  { _id: 'Goodyear', hotelsCounter: 9 },
  { _id: 'Surprise', hotelsCounter: 10 },
  { _id: 'Peoria', hotelsCounter: 12 },
  { _id: 'North Las Vegas', hotelsCounter: 12 },
  { _id: 'Glendale', hotelsCounter: 20 },
  { _id: 'Chandler', hotelsCounter: 30 },
  { _id: 'Henderson', hotelsCounter: 41 },
  { _id: 'Mesa', hotelsCounter: 53 },
  { _id: 'Tempe', hotelsCounter: 57 },
  { _id: 'Madison', hotelsCounter: 67 }
]
Type "it" for more
```

1c)



- ▶ JS
- ▶ reviewDB
- ▼ tipDB
  - | tip
- ▶ userDB

The screenshot shows the 'tipDB.tip' collection in MongoDB Compass. The 'Find' tab is selected. A 'Filter' input field contains the query '{ field: 'value' }'. Below it, a table displays 'QUERY RESULTS: 1-20 OF MANY' with one row of data:

_id	user_id	text	business_id	likes	date	type
ObjectId('6477b90e79d66f461d4141b1')	"Vefj29mjork1DLhALLNAsg"	"Great food, huge portions and a gift shop and showers."	"JwUE5GmEO-sH1FuwJgKB1Q"	0	"2012-05-16"	"tip"

```
use('tipDB');

db.tip.aggregate(
{
  $match: {date: /2012/}
},
{
  $group: {
    _id: "$business_id",
    count: {$sum: 1}
  }
},
{
  $sort: {count: -1}
})
```

```
Type "it" for more
Atlas atlas-84ca13-shard-0 [primary] tipDB> db.tip.aggregate(
...   {
...     $match: {date: /2012/}
...   },
...   {
...     $group: {
...       _id: "$business_id",
...       count: {$sum: 1}
...     }
...   },
...   {
...     $sort: {count: -1}
...   }
[... )
[
  { _id: 'jf67Z1pnwElRSXllpQHiJg', count: 1084 },
  { _id: 'hW0Ne_HTHEAgGF1rAdmR-g', count: 622 },
  { _id: '2e2e7WgqU1BnpxmQL5jbfw', count: 430 },
  { _id: 'CsNOg-u_wCuXSt9Z-xU92Q', count: 374 },
  { _id: 'AtjsjFzalWqJ7S9DUFQ4bw', count: 351 },
  { _id: 'zt1TpTuJ6y9n551sw9TaEg', count: 347 },
  { _id: 'AeKTQBtPRDHLAFL9bzbUnA', count: 258 },
  { _id: 'FV16IeXJp2W6pnghTz2FAw', count: 252 },
  { _id: 'eq6lQI039SBLC6sHm3idGA', count: 239 },
  { _id: 'Ht8mXLuqJSTPrU9kvzosUA', count: 227 },
  { _id: 'DjOxXobyGDwWt89q4z1twg', count: 221 },
  { _id: 'UZbkczu-KJour2Hgt_5WWw', count: 209 },
  { _id: 'JpHE7yhMS5ehA9e8WG_ETg', count: 208 },
  { _id: '8buIr1zBC070EcAQSZko7w', count: 201 },
  { _id: 'vxxMqBaAHuWdx4impsLSSA', count: 197 },
  { _id: 'Xhg93cMdemu5pAMkDoEdtQ', count: 197 },
  { _id: '4bEjOyTaDG24SY5TxsaUNQ', count: 196 },
  { _id: 'H_SuH7uLiYahDMbNBB9kog', count: 185 },
  { _id: 'z3SyT8blMIhsZNvKJgKcRA', count: 184 },
  { _id: 'eWPFXL1Bmu1ImtIa2Rqliw', count: 184 }
]
```

1d)

The screenshot shows the MongoDB Compass interface. On the left sidebar, there are buttons for 'Create Database', 'View Sample Datasets', and a search bar for namespaces. Below these are sections for 'JS', 'reviewDB' (which is expanded to show 'review1'), 'tipDB', and 'userDB'. The main area is titled 'reviewDB.review' and displays storage details: STORAGE SIZE: 138.62MB, LOGICAL DATA SIZE: 220.49MB, TOTAL DOCUMENTS: 256000, INDEXES TOTAL SIZE: 7.2MB. It includes tabs for 'Find', 'Indexes', 'Schema Anti-Patterns' (with 0 issues), 'Aggregation', and 'Search Indexes'. A 'Filter' input field with placeholder 'Type a query: { field: 'value' }' and a 'Reset' button are also present. The results section is titled 'QUERY RESULTS: 1-20 OF MANY' and shows a single document snippet:

```
_id: ObjectId('6477bd519bb3037b0fe7e426')
votes: Object
user_... : "zvJCrpm2y0ZrxKffwGQLA"
review_... : "ebcN2aqmNUuYNoyvQErgnA"
stars: 4
date : "2012-05-15"
text : "Got a letter in the mail last week that said Dr. Goldberg is moving to..."
type : "review"
business_... : "vcNAWiLM4dR7D2nwJ7nCA"
```

```
use('reviewDB');

db.review1.aggregate([
  {
    $group: {
      _id: null,
      cool: {
        $sum: {
          $cond: [{ $gt: ["$votes.cool", 0]}, 1, 0]
        }
      },
      useful: {
        $sum: {
          $cond: [{ $gt: ["$votes.useful", 0]}, 1, 0]
        }
      },
      funny: {
        $sum: {
          $cond: [{ $gt: ["$votes.funny", 0]}, 1, 0]
        }
      }
    },
    {
      $project: {
        _id: 0,
        cool: 1,
        useful: 1,
        funny: 1
      }
    }
])
```

```
Atlas atlas-84ca13-shard-0 [primary] reviewDB> db.review1.aggregate([
...   {
...     $group: {
...       _id: null,
...       cool: {
...         $sum: {
...           $cond: [{${gt: ["$votes.cool", 0]}}, 1, 0]
...         }
...       },
...       useful: {
...         $sum: {
...           $cond: [{${gt: ["$votes.useful", 0]}}, 1, 0]
...         }
...       },
...       funny: {
...         $sum: {
...           $cond: [{${gt: ["$votes.funny", 0]}}, 1, 0]
...         }
...       }
...     },
...   },
...   {
...     $project: {
...       _id: 0,
...       cool: 1,
...       useful: 1,
...       funny: 1
...     }
...   }
... ])
[ { cool: 77638, useful: 125472, funny: 62015 } ]
```

+ Create Database

View Sample Datasets

Search Namespaces

JS

reviewDB

tipDB

userDB

  | user

## userDB.user

STORAGE SIZE: 79.74MB LOGICAL DATA SIZE: 122.79MB TOTAL DOC

Find Indexes Schema Anti-Patterns 0

Filter Type a query: { field: 'value' }

QUERY RESULTS: 1-20 OF MANY

```
_id: ObjectId('6477c087e3b2715a3184c97a')
yelping_sin... : "2010-09"
▶ votes: Object
  review_cou... : 22
  na... : "Jasmine"
  user_... : "0vscrHoajVRa1Yk19XWdwA"
▶ frien... : Array
  fa... : 1
  average_stam... : 5
```

```
use('userDB');

db.user.find(
{
  $and:[
    {"votes.funny": {$eq: 0}},
    {"votes.useful": {$eq: 0}}
  ],
  {
    _id: false,
    name: true,
    "votes.funny": true,
    "votes.useful": true
  }
}.sort({"name": 1})
```

```
Atlas atlas-84ca13-shard-0 [primary] userDB> db.user.find(
...   {
...     $and:[
...       {"votes.funny": {$eq: 0}},
...       {"votes.useful": {$eq: 0}}
...     ]
...   },
...   {
...     _id: false,
...     name: true,
...     "votes.funny": true,
...     "votes.useful": true
...   }
... ).sort({"name": 1})
[
  { votes: { funny: 0, useful: 0 }, name: 'Bernard' },
  { votes: { funny: 0, useful: 0 }, name: ',Maria' },
  { votes: { funny: 0, useful: 0 }, name: '006969123' },
  { votes: { funny: 0, useful: 0 }, name: 'A' },
  { votes: { funny: 0, useful: 0 }, name: 'A' },
  { votes: { funny: 0, useful: 0 }, name: 'A' },
  { votes: { funny: 0, useful: 0 }, name: 'A' },
  { votes: { funny: 0, useful: 0 }, name: 'A' },
  { votes: { funny: 0, useful: 0 }, name: 'A' },
  { votes: { funny: 0, useful: 0 }, name: 'A' },
  { votes: { funny: 0, useful: 0 }, name: 'A' },
  { votes: { funny: 0, useful: 0 }, name: 'A' },
  { votes: { funny: 0, useful: 0 }, name: 'A' },
  { votes: { funny: 0, useful: 0 }, name: 'A' },
  { votes: { funny: 0, useful: 0 }, name: 'A' },
  { votes: { funny: 0, useful: 0 }, name: 'A' },
  { votes: { funny: 0, useful: 0 }, name: 'A' },
  { votes: { funny: 0, useful: 0 }, name: 'A' },
  { votes: { funny: 0, useful: 0 }, name: 'A' },
  { votes: { funny: 0, useful: 0 }, name: 'A' },
  { votes: { funny: 0, useful: 0 }, name: 'A' }
]
```

```
use('reviewDB');

db.review1.aggregate([
  {
    $group: {
      _id: "$business_id",
      avg: {$avg: "$stars"}
    }
  },
  {
    $match: {
      avg: {$gt: 3}
    }
  },
  {
    $sort: {
      _id: 1
    }
  }
])
```

```
Atlas atlas-84ca13-shard-0 [primary] reviewDB> db.review1.aggregate([
...   {
...     $group: {
...       _id: "$business_id",
...       avg: {$avg: "$stars"}
...     }
...   },
...   {
...     $match:{
...       avg: {$gt: 3}
...     }
...   },
...   {
...     $sort: {
...       _id: 1
...     }
...   }
... ])
```

```
[
```

```
{ _id: '--XBxR1D92RaV6TyUnP80w', avg: 3.6666666666666665 },
{ _id: '-0HGqwlfw3I8nkJyMHxAsQ', avg: 4 },
{ _id: '-0QBrNvhrPQCaeo7mTo0zQ', avg: 4.333333333333333 },
{ _id: '-1b0b2izeJBZjHC7NWxiPA', avg: 3.824324324324324 },
{ _id: '-34jE_5dujSMIOBudQsiQ', avg: 4.75 },
{ _id: '-3JXOT-i2gDbASLgDeOSwQ', avg: 4.75 },
{ _id: '-3WVw1TNQbPBzaKCaQQ1AQ', avg: 3.7892156862745097 },
{ _id: '-3qWPkQAxsggQJYqgi5myA', avg: 3.25 },
{ _id: '-3xbryp44xhpN4BohxXDdQ', avg: 3.8151260504201683 },
{ _id: '-4BtfPW3_v092G8pqHrv4w', avg: 3.625 },
{ _id: '-4zMr3jk0ykmsk5sxsQM1A', avg: 4 },
{ _id: '-5IgAihccTd8_iENVgpd9A', avg: 3.5 },
{ _id: '-5bLhMLNGBIVyoPOVl0yLg', avg: 3.333333333333335 },
{ _id: '-63VfA2tnYyzwRt81B1AKw', avg: 4.0588235294117645 },
{ _id: '-65rmLRQ5JDwI-l8UDEV7Q', avg: 3.8 },
{ _id: '-6Ft3hulif7O2sIdKiG0Og', avg: 4.5 },
{ _id: '-6053B-ksqSKzWM6Y9moEQ', avg: 3.25 },
{ _id: '-6Roo-EHgSdUa4rP3tWyRw', avg: 3.4571428571428573 },
{ _id: '-6c7Tfec-X0aDmzjK06R-Q', avg: 3.1666666666666665 },
{ _id: '-7DxAxnXkALTUrtyoXAg', avg: 4.6666666666666667 }
```

```
]
```

1f 2)

- [+ Create Database](#)
- [View Sample Datasets](#)
- [Search Namespaces](#)

▶ JS

▼ reviewDB

business

review1

```
use('reviewDB');

db.review1.aggregate([
  {
    $limit: 200
  },
  {
    $lookup: {
      from: "business",
      localField: "business_id",
      foreignField: "business_id",
      as: "company"
    }
  },
  {
    $unwind: "$company"
  },
  {
    $group: {
      _id: "$company.name",
      avg: {$avg: "$stars"}
    }
  },
  {
    $match: {
      avg: {$gt: 3}
    }
  },
  {
    $sort: {
      _id: 1
    }
  }
])
```

## reviewDB

LOGICAL DATA SIZE: 251.98MB ST

Collection Name	Documents
business	42153
review1	256000

```
Atlas atlas-84ca13-shard-0 [primary] reviewDB> db.review1.aggregate([
...   {
...     $limit: 200
...   },
...   {
...     $lookup: {
...       from: "business",
...       localField: "business_id",
...       foreignField: "business_id",
...       as: "company"
...     }
...   },
...   {
...     $unwind: "$company"
...   },
...   {
...     $group: {
...       _id: "$company.name",
...       avg: {$avg: "$stars"}
...     }
...   },
...   {
...     $match: {
...       avg: {$gt: 3}
...     }
...   },
...   {
...     $sort: {
...       _id: 1
...     }
...   }
... ])
[{"_id": "Beach House Restaurant & Lounge", "avg": 3.3870967741935485}, {"_id": "Chang Jiang Chinese Kitchen", "avg": 4}, {"_id": "Crandalls Carryout & Catering", "avg": 4}, {"_id": "Culver's", "avg": 4.375}, {"_id": "Deforest Family Restaurant", "avg": 3.923076923076923}, {"_id": "Eric Goldberg, MD", "avg": 3.75}, {"_id": "Green Lantern Restaurant", "avg": 3.8666666666666667}, {"_id": "Imperial Garden Chinese Restaurant", "avg": 3.2432432432432434}, {"_id": "Main Moon Chinese Restaurant", "avg": 3.5714285714285716}, {"_id": "Parsonage Bed & Breakfast the", "avg": 4.75}, {"_id": "Pine Cone Restaurant", "avg": 4.181818181818182}, {"_id": "Spartan Animal Hospital", "avg": 4.5}, {"_id": "Spartan Pizza", "avg": 3.4}]
```

## Zadanie 2

### **Embedded Data Model – podejście dokumentowe:**

Dane są hierarchicznie zagnieżdżane w jednym dokumencie, nie ma podziału na kolekcje.

#### Zalety:

- szybki odczyt danych o całym dokumencie
- większa wydajność (wystarczy jedno zapytanie bez konieczności łączenia dokumentów i kolekcji)

#### Wady:

- konieczność wykonywania bardziej złożonych zapytań, gdy chce się dostać do głęboko zagnieżdzonego dokumentu
- przy aktualizacji danych może wystąpić konieczność aktualizacji ich w wielu dokumentach, co negatywnie wpływa na wydajność

### **Referenced Data Model – podejście tabelaryczne:**

W dokumentach przechowywane są identyfikatory powiązanych dokumentów, a dane są podzielone na kolekcje.

#### Zalety:

- odseparowanie danych między kolekcjami daje elastyczność i skalowalność bazy danych
- dane są bardziej uporządkowane, dzięki czemu baza jest bardziej przejrzysta
- szybsza aktualizacja i pobieranie danych
- mniejsze ryzyko przekroczenia maksymalnego rozmiaru pojedynczego dokumentu, gdyż dane można rozdzielić na wiele dokumentów

#### Wady:

- konieczność łączenia dokumentów znajdujących się w różnych kolekcjach
- prawdopodobieństwo redundancji danych i błędów referencyjnych

## Wariant B - wycieczki, Referenced Data Model

Baza zawiera 4 kolekcje:

- firmy
- wycieczki
- osoby
- recenzje

The screenshot shows a MongoDB interface with the following elements:

- Top Bar:** + Create Database, View Sample Datasets, Search Namespaces.
- Left Sidebar:** ToursDB (expanded), Companies (selected), People, Reviews, Tours.
- Right Panel Header:** ToursDB.Companies, STORAGE SIZE: 4KB, LOGICAL DATA SIZE: 0B, TOTAL DOCUMENTS: 0.
- Right Panel Buttons:** Find, Indexes, Schema Anti-Patterns (0), Actions.
- Right Panel Input:** Filter (with icon), Type a query: { field: 'value' }.
- Right Panel Status:** QUERY RESULTS: 0.

Przykładowe dane wprowadzone do bazy:

```
const firstCompanyInsert = db.Companies.insertOne({
  name: "Trivago",
  country: "Poland",
  city: "Kraków",
  address: "Miła 12",
  phone: "123456789",
  email: "trivago@gmail.com",
  tours: []
});

const secondCompanyInsert = db.Companies.insertOne({
  name: "Booking",
  country: "Poland",
  city: "Warszawa",
  address: "Zła 1",
  phone: "987654321",
  email: "booking@gmail.com",
  tours: []
});

const firstPersonInsert = db.People.insertOne({
  name: "Jan Kowalski"
});

const secondPersonInsert = db.People.insertOne({
  name: "Anna Panna"
});

firstPerson = db.People.findOne(
  {_id: firstPersonInsert.insertedId}
)

secondPerson = db.People.findOne(
  {_id: secondPersonInsert.insertedId}
)
```

```
firstReview = db.Reviews.insertOne({
  stars: 5,
  user: firstPersonInsert.insertedId,
  comment: "Very nice."
});

secondReview = db.Reviews.insertOne({
  stars: 2,
  user: secondPersonInsert.insertedId,
  comment: "Not nice."
});

const firstTourInsert = db.Tours.insertOne({
  company: firstCompanyInsert.insertedId,
  tourName: "Hot n'Cold",
  price: 92000,
  destination: "Egypt and North Pole",
  availablePlaces: 7,
  startDate: "12.07.2023",
  endDate: "20.08.2023",
  reviews: [],
  tourists: []
});

const secondTourInsert = db.Tours.insertOne({
  company: secondCompanyInsert.insertedId,
  tourName: "Colorful Rainbow",
  price: 19800,
  destination: "New Zealand",
  availablePlaces: 2,
  startDate: "12.07.2023",
  endDate: "25.07.2023",
  reviews: [],
  tourists: []
});
```

```

firstTour = db.Tours.findOne(
|  {_id: firstTourInsert.insertedId}
|
secondTour = db.Tours.findOne(
|  {_id: secondTourInsert.insertedId}
|)

db.Companies.updateOne(
|  {_id: firstCompanyInsert.insertedId},
|  {$push: {tours: firstTourInsert.insertedId}}
|)

db.Companies.updateOne([
|  {_id: secondCompanyInsert.insertedId},
|  {$push: {tours: secondTourInsert.insertedId}}
|])

db.Tours.updateOne(
|  {_id: firstTourInsert.insertedId},
|  {$push: {reviews: firstReview.insertedId, tourists: firstPersonInsert.insertedId}}
|)

db.Tours.updateOne(
|  {_id: secondTourInsert.insertedId},
|  {$push: {reviews: secondReview.insertedId, tourists: secondPersonInsert.insertedId}}
|)
```

The screenshot shows the MongoDB Compass interface with the following details:

- Left Sidebar:**
  - + Create Database
  - View Sample Datasets
  - Search Namespaces
  - ToursDB**
    - Companies
    - People** (highlighted in green)
    - Reviews
    - Tours
- Top Right Panel:**

**ToursDB.People**

STORAGE SIZE: 36KB   LOGICAL DATA SIZE: 176B   TOTAL DOCUMENTS: 4   II

Find   Indexes   Schema Anti-Patterns 0   Aggre
- Bottom Panel:**

Filter  Type a query: { field: 'value' }

QUERY RESULTS: 1-2 OF 2

  - \_id: ObjectId('64790d1189e77d823af851b9')  
na... : "Jan Kowalski"
  - \_id: ObjectId('64790d1289e77d823af851ba')  
na... : "Anna Panna"

+ Create Database

View Sample Datasets

Search Namespaces

▼ ToursDB

Companies

People

Reviews

Tours

## ToursDB.Reviews

STORAGE SIZE: 36KB LOGICAL DATA SIZE: 298B TOTAL DOCUMENTS

Find

Indexes

Schema Anti-Patterns 0

Filter 

Type a query: { field: 'value' }

### QUERY RESULTS: 1-2 OF 2

```
_id: ObjectId('64790d1289e77d823af851bb')
stars: 5
us... : ObjectId('64790d1189e77d823af851b9')
comme... : "Very nice."
```

```
_id: ObjectId('64790d1289e77d823af851bc')
stars: 2
us... : ObjectId('64790d1289e77d823af851ba')
comme... : "Not nice."
```

+ Create Database

View Sample Datasets

Search Namespaces

**ToursDB.Companies**

STORAGE SIZE: 36KB LOGICAL DATA SIZE: 353B TOTAL DOCUMENTS: 2

Find Indexes Schema Anti-Patterns 0 Agg

Filter Type a query: { field: 'value' }

QUERY RESULTS: 1-2 OF 2

```
_id: ObjectId('647a3ed889e77d823af851f7')
na... : "Trivago"
count... : "Poland"
ci... : "Kraków"
addre... : "Miła 12"
phone: "123456789"
email: "trivago@gmail.com"
▼ tours: Array
  0: ObjectId('647a3eda89e77d823af851fd')
```

+ Create Database

View Sample Datasets

Search Namespaces

**ToursDB.Companies**

STORAGE SIZE: 4KB LOGICAL DATA SIZE: 427B TOTAL DOCUMENTS: 2 INDEX

Find Indexes Schema Anti-Patterns 0 Aggregati

Filter Type a query: { field: 'value' }

QUERY RESULTS: 1-2 OF 2

```
▶ _id: ObjectId('647a3ed889e77d823af851f7')
na... : "Trivago"
count... : "Poland"
ci... : "Kraków"
addre... : "Miła 12"
phone: "123456789"
email: "trivago@gmail.com"
▼ tours: Array
  ▼ 0: Object
    i...: ObjectId('647a3eda89e77d823af851fd')
    tourNa... : "Hot n'Cold"
```

+ Create Database

View Sample Datasets

Search Namespaces

**ToursDB.Tours**

STORAGE SIZE: 36KB LOGICAL DATA SIZE: 521B TOTAL DOCUMENTS: 2

**Find** Indexes Schema Anti-Patterns 0 Aggs

Filter Type a query: { field: 'value' }

**QUERY RESULTS: 1-2 OF 2**

```
_id: ObjectId('647a3eda89e77d823af851fd')
compa... : ObjectId('647a3ed889e77d823af851f7')
tourNa... : "Hot n'Cold"
price: 92000
destinati... : "Egypt and North Pole"
availablePlaces: 7
startDa... : "12.07.2023"
endDa... : "20.08.2023"
▶ review... : Array
▼ tourist... : Array
  0: ObjectId('647a3ed889e77d823af851f9')
```

*Wyszukiwanie wycieczek danej osoby:*

```
Atlas atlas-84ca13-shard-0 [primary] ToursDB> db.Tours.find(
...   {tourists: firstPersonInsert.insertedId}
[... )
[
  {
    _id: ObjectId("647a3eda89e77d823af851fd"),
    company: ObjectId("647a3ed889e77d823af851f7"),
    tourName: "Hot n'Cold",
    price: 92000,
    destination: 'Egypt and North Pole',
    availablePlaces: 7,
    startDate: '12.07.2023',
    endDate: '20.08.2023',
    reviews: [ ObjectId("647a3ed989e77d823af851fb") ],
    tourists: [ ObjectId("647a3ed889e77d823af851f9") ]
  }
]
```

*Wyszukiwanie recenzji danej osoby:*

```
Atlas atlas-84ca13-shard-0 [primary] ToursDB> db.People.aggregate([
...     {
...         $match: { _id: firstPersonInsert.insertedId }
...     },
...     {
...         $lookup: {
...             from: "Reviews",
...             localField: "_id",
...             foreignField: "user",
...             as: "reviewInfo"
...         }
...     },
...     {
...         $unwind: "$reviewInfo"
...     },
...     {
...         $lookup: {
...             from: "Tours",
...             localField: "_id",
...             foreignField: "tourists",
...             as: "tourInfo"
...         }
...     },
...     {
...         $unwind: "$tourInfo"
...     },
...     {
...         $project: {
...             personName: firstPerson.name,
...             personComment: "$reviewInfo.comment",
...             starsGiven: "$reviewInfo.stars",
...             tourName: "$tourInfo.tourName"
...         }
...     }
... ],
[
{
    _id: ObjectId("647a3ed889e77d823af851f9"),
    personName: 'Jan Kowalski',
    personComment: 'Very nice.',
    starsGiven: 5,
    tourName: "Hot n'Cold"
}
]
```

*Zmiana imienia i nazwiska osoby:*

Przed zmianą:

```
_id: ObjectId('647a3ed889e77d823af851f9')
na... : "Jan Kowalski"

Atlas atlas-84ca13-shard-0 [primary] ToursDB> db.People.updateOne(
...   {_id: firstPersonInsert.insertedId},
...   {$set: {name: "Grzegorz Pomidor"}}
[... )
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

Po zmianie:

```
_id: ObjectId('647a3ed889e77d823af851f9')
na... : "Grzegorz Pomidor"
```

```

Atlas atlas-84ca13-shard-0 [primary] ToursDB> firstPerson = db.People.findOne(
...     {_id: firstPersonInsert.insertedId}
...   )
{ _id: ObjectId("647a3ed889e77d823af851f9"), name: 'Grzegorz Pomidor' }
Atlas atlas-84ca13-shard-0 [primary] ToursDB>

Atlas atlas-84ca13-shard-0 [primary] ToursDB> db.People.aggregate([
...   {
...     $match: { _id: firstPersonInsert.insertedId}
...   },
...   {
...     $lookup:{
...       from: "Reviews",
...       localField: "_id",
...       foreignField: "user",
...       as: "reviewInfo"
...     }
...   },
...   {
...     $unwind: "$reviewInfo"
...   },
...   {
...     $lookup:{ 
...       from: "Tours",
...       localField: "_id",
...       foreignField: "tourists",
...       as: "tourInfo"
...     }
...   },
...   {
...     $unwind: "$tourInfo"
...   },
...   {
...     $project: {
...       personName: firstPerson.name,
...       personComment: "$reviewInfo.comment",
...       starsGiven: "$reviewInfo.stars",
...       tourName: "$tourInfo.tourName"
...     }
...   }
... ])
[
{
  _id: ObjectId("647a3ed889e77d823af851f9"),
  personName: 'Grzegorz Pomidor',
  personComment: 'Very nice.',
  starsGiven: 5,
  tourName: "Hot n'Cold"
}
]

```

Wystarczyło zmienić imię tylko w jednym miejscu, aby zmiana pojawiła się przy zapytaniach.

## Wariant B - wycieczki, Embedded Data Model

Baza zawiera 2 kolekcje:

- wycieczki (zawierają informacje o wycieczkach, ich recenzjach i biurach, które je organizują)
- osoby (zawierają informacje o osobach)

The screenshot shows the MongoDB Compass interface. On the left, there's a sidebar with buttons for 'Create Database', 'View Sample Datasets', and 'Search Namespaces'. Below that is a tree view of databases and collections: 'ToursDB' is expanded, showing 'People' and 'Tours'. 'People' is further expanded, showing 'Filter' and a search bar with the placeholder 'Type a query: { field: 'value'}'. At the bottom of the interface, it says 'QUERY RESULTS: 0'.

+ Create Database

View Sample Datasets

Search Namespaces

ToursDB

People

Tours

Filter Type a query: { field: 'value'}

QUERY RESULTS: 0

Dodanie przykładowych danych:

```
firstTourInsert = db.Tours.insertOne({
  companyInfo: {
    companyName: "Trivago",
    country: "Poland",
    city: "Kraków",
    address: "Miła 12",
    phone: "123456789",
    email: "trivago@gmail.com"
  },
  reviews: [],
  tourName: "Hot n'Cold",
  price: 92000,
  destination: "Egypt and North Pole",
  availablePlaces: 7,
  startDate: "12.07.2023",
  endDate: "20.08.2023",
});

secondTourInsert = db.Tours.insertOne({
  companyInfo: {
    companyName: "Booking",
    country: "Poland",
    city: "Warszawa",
    address: "Zła 1",
    phone: "987654321",
    email: "booking@gmail.com",
  },
  reviews: [],
  tourName: "Colorful Rainbow",
  price: 19800,
  destination: "New Zealand",
  availablePlaces: 2,
  startDate: "12.07.2023",
  endDate: "25.07.2023",
});
```

```

const firstPersonInsert = db.People.insertOne({
  name: "Jan Kowalski"
});
const secondPersonInsert = db.People.insertOne({
  name: "Anna Panna"
});

firstPerson = db.People.findOne(
  {_id: firstPersonInsert.insertedId}
)

secondPerson = db.People.findOne(
  {_id: secondPersonInsert.insertedId}
)

```

The screenshot shows the MongoDB Compass interface. On the left, there's a sidebar with buttons for 'Create Database', 'View Sample Datasets', and a search bar for namespaces. Below that, under 'ToursDB', there are two collections: 'People' and 'Tours'. The 'Tours' collection is currently selected, indicated by a green border.

The main area displays the 'ToursDB.Tours' collection. At the top, it shows storage details: STORAGE SIZE: 20KB, LOGICAL DATA SIZE: 664B, and TOTAL DOCUMENTS: 2. Below this, there are tabs for 'Find', 'Indexes', 'Schema Anti-Patterns (0)', and 'Aggregations'.

A search bar at the top right contains the placeholder 'Type a query: { field: 'value' }'. The results section shows one document:

```

_id: ObjectId('647a56aa89e77d823af85201')
companyIn... : Object
  companyNa... : "Trivago"
  count... : "Poland"
  ci... : "Kraków"
  addre... : "Miła 12"
  phone: "123456789"
  email: "trivago@gmail.com"
review... : Array
  tourNa... : "Hot n'Cold"
  price: 92000
  destinati... : "Egypt and North Pole"
  availablePlaces: 7
  startDa... : "12.07.2023"
  endDa... : "20.08.2023"

```

+ Create Database

View Sample Datasets

Search Namespaces

ToursDB

People

Tours

ToursDB.People

STORAGE SIZE: 20KB LOGICAL DATA SIZE: 88B TOTAL DOCUMENTS: 88

Find Indexes Schema Anti-Patterns 0

Filter Type a query: { field: 'value' }

QUERY RESULTS: 1-2 OF 2

```
_id: ObjectId('647a567a89e77d823af851ff')
na... : "Jan Kowalski"
```

```
_id: ObjectId('647a567a89e77d823af85200')
na... : "Anna Panna"
```

Aktualizacja recenzji:

```
firstReview = {
  stars: 5,
  user: firstPersonInsert.insertedId,
  comment: "Very nice."
};

secondReview = {
  stars: 2,
  user: secondPersonInsert.insertedId,
  comment: "Not nice."
};

db.Tours.update(
  {_id: firstTourInsert.insertedId},
  {$push: {reviews: firstReview}}
)

db.Tours.update(
  {_id: secondTourInsert.insertedId},
  {$push: {reviews: secondReview}}
)
```

```

_id: ObjectId('647a56aa89e77d823af85201')
▶ companyIn... : Object
▼ review... : Array
  ▼ 0: Object
    stars: 5
    user... : ObjectId('647a567a89e77d823af851ff')
    comment... : "Very nice."
  tourName... : "Hot n'Cold"
  price: 92000
  destination... : "Egypt and North Pole"
  availablePlaces: 7
  startDa... : "12.07.2023"
  endDa... : "20.08.2023"

```

*Wyszukanie wycieczek danej osoby:*

(wcześniej jeszcze jedna opinia secondPerson została dodana do wycieczki numer 1)

```

Atlas atlas-84ca13-shard-0 [primary] ToursDB> db.People.aggregate([
...   {
...     $match: {"_id": secondPersonInsert.insertedId}
...   },
...   {
...     $lookup: {
...       from: "Tours",
...       localField: "_id",
...       foreignField: "reviews.user",
...       as: "info"
...     }
...   },
...   {
...     $unwind: "$info"
...   },
...   {
...     $project: {
...       personName: "$name",
...       tourName: "$info.tourName",
...       stars: "$info.review.stars"
...     }
...   }
... ])
[{"_id": ObjectId("647a567a89e77d823af85200"),
  personName: 'Anna Panna',
  tourName: "Hot n'Cold"}, {"_id": ObjectId("647a567a89e77d823af85200"),
  personName: 'Anna Panna',
  tourName: 'Colorful Rainbow'}]

```

*Wyszukanie recenzji danej osoby:*

```
Atlas atlas-84ca13-shard-0 [primary] ToursDB> db.Tours.find(
...   {
...     reviews:
...       {
...         $elemMatch: {user: secondPersonInsert.insertedId}
...       }
...     },
...   {
...     name: secondPerson.name,
...     tourName: true,
...     "reviews.$": 1
...   }
[... )
[
  {
    _id: ObjectId("647a56aa89e77d823af85201"),
    reviews: [
      {
        stars: 2,
        user: ObjectId("647a567a89e77d823af85200"),
        comment: 'Not nice.'
      }
    ],
    tourName: "Hot n'Cold",
    name: 'Anna Panna'
  },
  {
    _id: ObjectId("647a56ab89e77d823af85202"),
    reviews: [
      {
        stars: 2,
        user: ObjectId("647a567a89e77d823af85200"),
        comment: 'Not nice.'
      }
    ],
    tourName: 'Colorful Rainbow',
    name: 'Anna Panna'
  }
]
```