

University of Vienna
Information Management and System Engineering

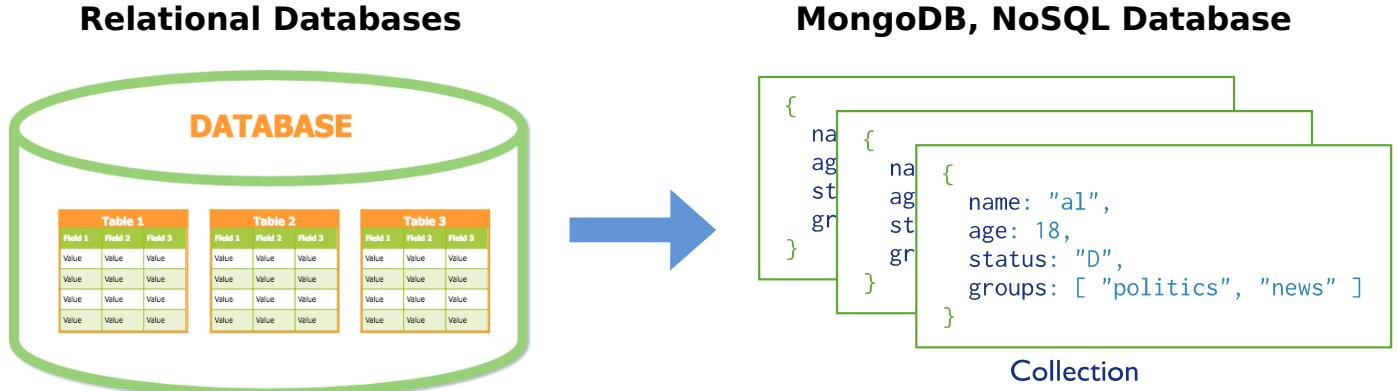
Semester Projekt Massage Praxis



Tornike Khachidze, 01469313

3.1. NoSQL Database design

Unsere Aufgabe bestand darin, soweit ich verstanden habe, dass wir unsere Sql Struktur, die wir in MS2 implementiert haben, in NoSQL zu transportieren.



MongoDB

- Ermöglicht einen schnelleren Zugriff auf die Daten, da der interne Speicher für die Speicherung verwendet wird.
- Die Fähigkeit, ein dokumentbasiertes Datenmodell abzuleiten.
- Vermeidung der Datenreplikation und damit Erreichung von Consistency & Partition Toleranz.
- Da es sich um eine NOSQL-Datenbank handelt, ist sie offensichtlich sicher, da keine SQL-Injektion durchgeführt werden kann
- Überdenken und Neugestalten der Beziehungen zwischen den Objekten.
- Formal gesehen ist ein Dokument eine geordnete Menge von Key-value-Paaren, wobei ein value ein einfacher Datentyp, ein Array von Values oder ein eingebettetes Dokument sein kann.
- Die Schemafreiheit bietet aber auch eine bislang ungewohnte Flexibilität: Daher können so Objekte mit 1:n-Beziehungen direkt in einem Datensatz(Dokument) abspeichert werden.
- Bei relationalen Datenbanken liegt der Fokus bei der Datenmodellierung traditionell auf der effizienten Ablage der Daten. Im Gegensatz dazu muss beim Schema-Design in MongoDB(NoSQL) der Fokus auf den Abfragen liegen, die auf den Daten ausgeführt werden sollen.

Daher habe ich MongoDB ausgewählt. Meiner Meinung nach ist MongoDB für meiner Projekt nützlich und sinnvoll.

NoSQL Design

Arzt

```
{  
  "ArztID": 1,  
  "aNachname": "Lennon",  
  "aVorname": "John",  
  "aFachgebiet": "Musicmaster",  
  "Patient": [
```

```
  {  
    "PatienID": 1,  
    "pNachname": "Musiclooser",  
    "pVorname": "Stive",  
    "Adresse": {  
      "PLZ": "2500",  
      "Ort": "Wien",  
      "Strasse": "Liverpoolgasse"  
    },  
    "Gebdat": "10/09/1990"  
  },
```

```
{  
  "PatienID": 2,  
  "pNachname": "Musicfan",  
  "pVorname": "George",  
  "Adresse": {  
    "PLZ": "2100",  
    "Ort": "Wien",  
    "Strasse": "MusicGasse"  
  },  
  "Gebdat": "10/10/1995"  
},
```

Usw.....
]
}

In Arzt werden alle Patienten gespeichert die von einem bestimmten Arzt eine Verordnung haben. z.B in System unter Arzt mit id 1 werden auch alle Patienten gespeichert, die ihre Verordnung von Arzt 1 bekomme haben. Da Patienten nicht sehr oft geändert werden, so wird Zeit für die Abfrage gespart.

Heilmasseur

{

“HeilmasseurID”: 1,
“hNachname”: “Fischer”,
“hVorname”: “Debora”,
“T-nummer”: “0676/6578909”,
“Alt”: “45”,
“leitet” [],
“Patient” : [

{

“PatienID”: 1,
“pNachname”: “Musiclooser”,
“pVorname”: “Stive”,
“Adresse” : {
“PLZ” : “2500”,
“Ort” : “Wien”,
“Strasse” : “Liverpoolgasse”
},
“Gebdat”: “10/09/1990”
},

{

“PatienID”: 2,
“pNachname”: “Musicfan”,
“pVorname”: “George”,
“Adresse” : {
“PLZ” : “2100”,

```
"Ort" : "Wien",
"Strasse" : "MusicGasse"
},
"Gebdat": "10/10/1995"
},
```

1,

Praxisgemeinschaft

{

```
"PgID": 1,
"pgname": "Massage Praxis Baden",
"Praxisraum": [
```

```
{
  "Raumnummer": 02,
  "prName": "Heilmassage"
},
```

```
{
  "Raumnummer": 03,
  "prName": "FussMassage"
},
```

Usw....

]

}

}

Hier habe ich so gemeint, dass der Heilmasseur mehrere Patienten hat, auch die Praxisgemeinschaft, wo Heilmasseur arbeitet und die Praxisgemeinschaft hat mehrere Räume... Also speichert Patienten in Heilmasseur und auch Praxisgemeinschaft, wo auch Praxisräume gespeichert werden. Dadurch wird meiner Meinung nach die Abfrage beschleunigt.

Behandlung

```
{  
    "HeilmasseurID": 1,  
    "Patient": [  
        {
```

```
            "PatienID": 96,  
            "BhDatum": "2015-04-09",  
            "BArt": "Segmentmassage",  
            "BPreis": 79  
        },
```

```
        {  
            "PatienID": 43,  
            "BhDatum": "2015-11-14",  
            "BArt": "Heilmassage",  
            "BPreis": 88  
        },
```

],

"Praxisgemeinschaft"

```
{  
    "PgID": 1,  
    "pgname": "HeilmassageRaum",  
    "Praxisraum": 2  
}
```

}

Speichert sowohl Patienten als Array values als auch Praxisgemeinschaft als eingebettetes Dokument in Behandlung, um Abfrage schneller zu erhalten.

Patient

{

```
  "PatienID": 5,  
    "pNachname": "Georg",  
    "pVorname": "Harisson",  
    "Adresse" : {  
      "PLZ" : "2501",  
      "Ort" : "Baden",  
      "Strasse" : "Coolgasse"  
    },  
    "Gebdat": "10/09/1982",  
    "Privat" : {  
      "Rechnungsnummer": 1011,  
      "Rechnungsdatum": "10/15/2016",  
      "Summe": 100,  
      "Begründung": "Kein Versicherung:")"  
      "Betrag": 55  
    },  
  },  
}
```

}

Speichert die eingebetteten Dokumente(Rechnung[Privat] und Adresse) in Patient. Das beschleunigt die Abfrage.

Patient

{

```
  "PatienID": 5,  
    "pNachname": "Georg",  
    "pVorname": "Harisson",  
    "Adresse" : {  
      "PLZ" : "2501",  
      "Ort" : "Baden",  
      "Strasse" : "Coolgasse"  
    },  
    "Gebdat": "10/09/1982",  
    "GKV" : {  
      "Rechnungsnummer": 1011,  
    },  
  },  
}
```

"Rechnungsdatum": "10/15/2016",

"Summe": 100,

"T-nummer": "0676/7778990",

"Website": "www.gkv.at",

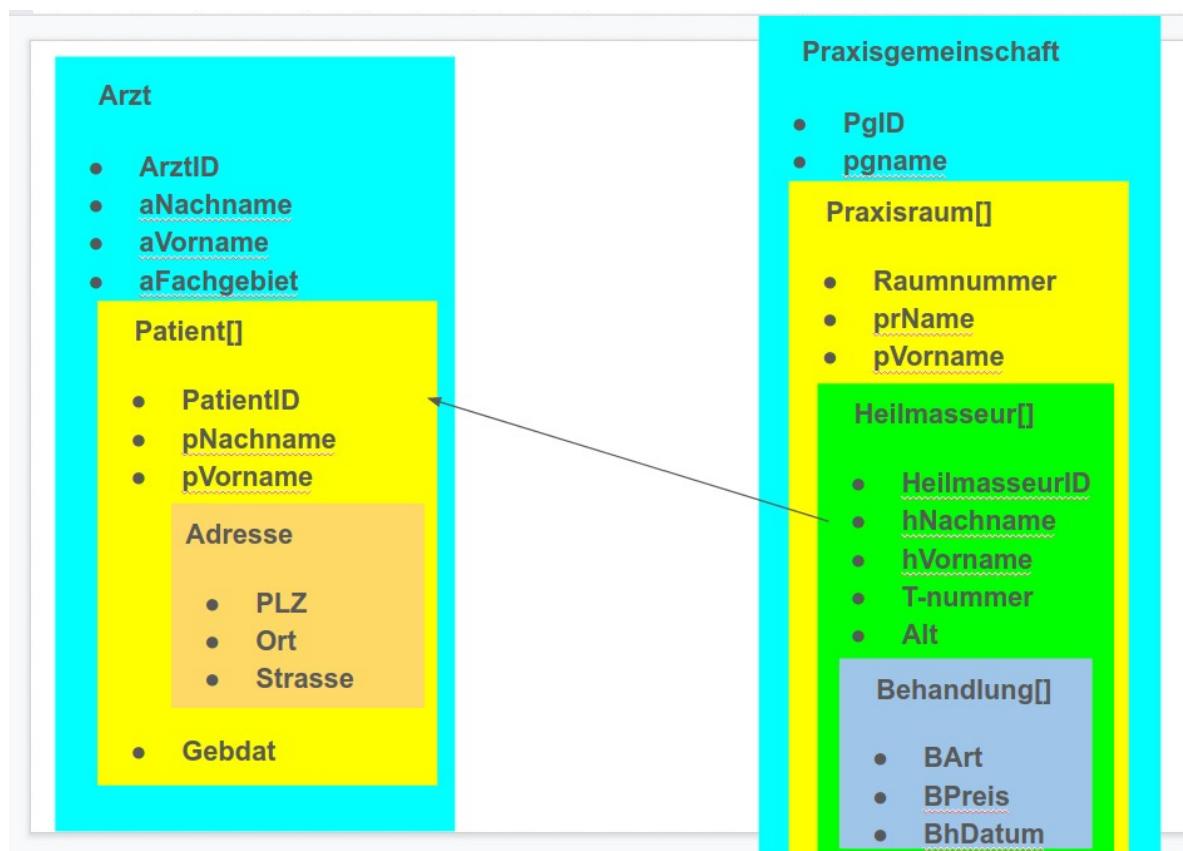
"Fax": "+4314000-89889"

}

}

Speichert die eingebetteten Dokumente(Rechnung[GKV] und Adresse) in Patient. Das beschleunigt die Abfrage.

3.1.2 Alternatives Modell



Hier geht es um eine Alternative. Bei Arzt ändert es fast nichts, doch andere könnten so dargestellt werden.

Praxisgemeinschaft → speichert Praxisraum, Heilmasseur und Behandlung in Praxisgemeinschaft und damit kann auch die Abfrage beschleunigt werden. Die Logik dabei ist, dass in der Praxisgemeinschaft mehrere Praxisräume existieren, wo mehrere Heilmasseure arbeiten und Behandlungen von Heilmasseuren durchgeführt werden.



Speichert Rechnung in Patient, es werden mehrere Rechnungen in Patient gespeichert werden, die entweder GKV oder Privat sein könnten.

3.1.3 Indexing and discussion on optimized sharding of NoSQL DBMS for the IS

MongoDB verfügt über ein umfangreiches Abfragemodell, das Flexibilität beim Zugriff auf Daten bietet. Standardmäßig erstellt MongoDB einen Index für das Primärschlüsselfeld `_id` des Dokuments.

Alle benutzerdefinierten Indizes sind Sekundärindizes. Jedes Feld kann für einen Sekundärindex verwendet werden, einschließlich der Felder in Unterdokumenten und Arrays.

Index options for MongoDB include:

Zusammengesetzte Indizes: Verwenden der Indexschnittstelle MongoDB kann mehr als einen Index verwenden, um eine Abfrage zu erfüllen. Diese Funktion ist nützlich, wenn Ad-hoc-Abfragen ausgeführt werden, da Datenzugriffsmuster normalerweise nicht im Voraus bekannt sind.

Eindeutige Indizes: By specifying an index as unique, MongoDB will reject inserts of new documents or updates to existing documents which would have resulted in duplicate values for the indexed field.

Array-Indizes: Für Felder, die ein Array enthalten, wird jeder Array-Wert als separater Indexeintrag gespeichert.

TTL-Indizes: In einigen Fällen sollten Daten verfallen automatisch.

Geodatenverzeichnisse: MongoDB stellt Geodatenindizes zur Verfügung, um Abfragen in Bezug auf den Standort in einem zweidimensionalen Raum zu optimieren, z. B. Projektionssysteme für die Erde.

Utc... Sparse, Teilweise , Hash, Textsuche, Indizes.

Indizes können für jeden Teil des JSON-Dokuments erstellt werden - einschließlich Unterdokumenten und Array-Elementen - und sind damit viel leistungsfähiger als die von RDBMS angebotenen

Indizes werden verwendet , um die Leistung für allgemeine Abfragen zu verbessern.

Sharding

Ist eine Methode zum Verteilen von Daten auf mehrere Computer. MongoDB verwendet Sharding, um Bereitstellungen mit sehr großen Datenmengen und hohem Durchsatz zu unterstützen.

Wenn man ihre Schreib und Lesezugriffe horizontal skalieren möchte, verwendet man sogenante Sharding. Dabei werden die Dokumente anhand eines Shard Keys disjunkt auf mehrere Shards verteilt.

Ein Shard kann eine einzelne mongod-Instanz sein oder besser ein ganzes Replica Set. Der Client verbindet sich dann gegen Router-Prozesse, die anhand des Shard Keys die Schreib oder Lese Operation an den passenden Shard bzw. Eine Gruppe von Shards delegieren.

Um die Performance des Datenbanksystems zu steigern, wird in MongoDB das Sharding verwendet.

3.2 Data migration

Anmerkung:

Um in MS3 eine Migration durchzuführen, ist MS2 auch nötig. Daher lade ich auch MS2 mit.

Fakten:

- Docker mongodb Container (bitnami/mongodb:latest).
- Es sind keine Änderungen in MS2 durchgeführt worden.

Setup:

- Für die Daten Migration habe ich Java verwendet.
- Dataimport.jar wurde erstellt (java -jar Dataimport.jar) Mit Hilfe dieses jar werden die Datensätze von SQL Database rausgenommen und in NoSQL Database hinzugefügt, eigentlich Migration wird durchgeführt.

3.3 Implementation IS (NoSQL)

Realization of all Use-Cases from Milestone 2

Anmerkung:

MS2 --> [In dieser Website funktionieren nicht nur **Patient** und **Behandlung**, die ich für M3 implementiert habe, sondern alle anderen auch. Aber in diesem Bericht wurden nur zwei Use cases gezeigt, die für M3 notwendig sind.]. Daher habe ich in MS3 wieder die zwei Use cases **Patient** und **Behandlung** implementiert.

1. Use case Patient von MS2

Erster Use case in MS2 (der Arzt kann Patient im System anschauen, falls Patient vorhanden, Verordnung schreiben oder Daten aktualisieren, falls Patient nicht vorhanden, Patient im System hinzufügen).

In MS3 habe ich Arzt erstellt und in Arzt Array values documents("Patients"). Die Idee dahinter ist, dass jedem Arzt alle Patienten hinzugefügt werden, die von diesem Arzt eine Verordnung haben. Der Arzt kann Patient anschauen, aktualisieren, hinzufügen und löschen.

Arzt

```
{
  "_id" : ObjectId("5d0a10d7f578c3657b340abd"),
  "ArztID" : 1,
  "aNachname" : "Dina",
  "aVorname" : "Finkel",
  "aFachgebiet" : "NEUROLOGE",
  "patients" : [
    {
      "PatientID" : "45",
      "Nachname" : "Robben",
      "Vorname" : "Karsten",
      "PLZ" : "5544",
      "Ort" : "Milan",
      "Strasse" : "nordgasse",
      "Gdat" : "1977-04-10",
      "ArztID" : "1",
      "Diagnosse" : "diagnoss7",
      "ADatum" : "2015-09-25"
    },
    {
      "PatientID" : "94",
      "Nachname" : "Nesta",
      "Vorname" : "Roberto",
      "PLZ" : "2785",
      "Ort" : "Graz",
      "Strasse" : "ostgasse",
      "Gdat" : "1995-01-24",
      "ArztID" : "1",
      "Diagnosse" : "diagnoss7",
      "ADatum" : "2016-02-13"
    }
  ]
}
```

Hier sieht man z.B dass vom Arzt Dina Finkel mit ID 1 zwei Patienten eine Verordnung haben.

```
+ MS3_Khachidze_01469313 cd Dataimport
+ Dataimport java -jar Dataimport.jar
```

Nachdem die Migration erfolgreich durchgeführt worden ist, kann use case getestet werden.

```
client_crud.txt  createjs  Dataimport  docker-compose.yml  mongo_client
→ MS3_Khachidze_01469313 cd mongo_client
→ mongo_client java -jar mongo_client.jar Arzt read
```

Es wird überprüft, ob Datensätze in DB eingetragen sind.

```
name" : "Varabye", "aFachgebiet" : "NEUROLOGE", "patients" : [{ "PatientID" : "73", "Nachname" : "Vieri", "Vorname" : "Arien", "PLZ" : "7392", "Ort" : "Wien", "Strasse" : "Hochstrasse", "Gdat" : "1946-08-02", "ArztID" : "95", "Diagnosse" : "diagnoss8", "ADatum" : "2015-08-23" }]
}
[ { "_id" : { "$oid" : "5d0a10d8f578c3657b340b1c" }, "ArztID" : 96, "aNachname" : "Leonel", "aVorname" : "Varabye", "aFachgebiet" : "NEUROLOGE", "patients" : [{ "PatientID" : "82", "Nachname" : "Inzaghi", "Vorname" : "Roberto", "PLZ" : "5169", "Ort" : "Baden", "Strasse" : "tottistrasse", "Gdat" : "1942-05-23", "ArztID" : "96", "Diagnosse" : "diagnoss8", "ADatum" : "2015-1-19" }, { "PatientID" : "93", "Nachname" : "Inzaghi", "Vorname" : "Paolo", "PLZ" : "8809", "Ort" : "Birmingem", "Strasse" : "Lkvgasse", "Gdat" : "1990-01-29", "ArztID" : "96", "Diagnosse" : "diagnoss7", "ADatum" : "2015-01-21" }] }
]
[ { "_id" : { "$oid" : "5d0a10d8f578c3657b340b1d" }, "ArztID" : 97, "aNachname" : "Matias", "aVorname" : "Varabye", "aFachgebiet" : "NEUROLOGE", "patients" : [ ]
]
[ { "_id" : { "$oid" : "5d0a10d8f578c3657b340b1e" }, "ArztID" : 98, "aNachname" : "Andrey", "aVorname" : "Finkel", "aFachgebiet" : "NEUROLOGE", "patients" : [ ]
]
[ { "_id" : { "$oid" : "5d0a10d8f578c3657b340b1f" }, "ArztID" : 99, "aNachname" : "Ivan", "aVorname" : "Messi", "aFachgebiet" : "NEUROLOGE", "patients" : [{ "PatientID" : "31", "Nachname" : "Bagio", "Vorname" : "Alexsandro", "PLZ" : "4786", "Ort" : "Linz", "Strasse" : "ostgasse", "Gdat" : "1993-10-14", "ArztID" : "99", "Diagnosse" : "diagnoss10", "ADatum" : "2016-02-28" }]
]
[ { "_id" : { "$oid" : "5d0a10d8f578c3657b340b20" }, "ArztID" : 100, "aNachname" : "Leonel", "aVorname" : "Polaschek", "aFachgebiet" : "NEUROLOGE", "patients" : [ ]
]
→ mongo_client
→ mongo_client []
```

Alle Datensätze sind in DB.

Create Arzt

```
→ mongo_client java -jar mongo_client.jar Arzt create 1234 kul hakan Heilmasseur 101 Leonardo Dicaprio 1220 wien waehiring 05-07-1987 1234 krank 12-12-2019
Jun 19, 2019 1:39:43 PM com.mongodb.diagnostics.logging.JULLLogger log
INFO: Cluster created with settings {hosts=[localhost:27017], mode=SINGLE, requiredClusterType=UNKNOWN, serverSelectionTimeout='30000 ms', maxWaitQueueSize=500}
Arzt
Jun 19, 2019 1:39:44 PM com.mongodb.diagnostics.logging.JULLLogger log
INFO: No server chosen by WritableServerSelector from cluster description ClusterDescription{type=UNKNOWN, connectionMode=SINGLE, serverDescriptions=[ServerDescription{address=localhost:27017, type=UNKNOWN, state=CONNECTING}]]. Waiting for 30000 ms before timing out
Jun 19, 2019 1:39:44 PM com.mongodb.diagnostics.logging.JULLLogger log
INFO: Opened connection [connectionId{localValue:1, serverValue:29}] to localhost:27017
Jun 19, 2019 1:39:44 PM com.mongodb.diagnostics.logging.JULLLogger log
INFO: Monitor thread successfully connected to server with description ServerDescription{address=localhost:27017, type=STANDALONE, state=CONNECTED, ok=true, version=ServerVersion{versionList=[4, 0, 10]}, minWireVersion=0, maxWireVersion=7, maxDocumentSize=16777216, roundTripTimeNanos=4881205}
Jun 19, 2019 1:39:44 PM com.mongodb.diagnostics.logging.JULLLogger log
INFO: Opened connection [connectionId{localValue:2, serverValue:30}] to localhost:27017
create successful
→ mongo_client □
```

Hier wird ein neuer Arzt mit der ID 1234 erstellt.

```
}
```

 `"_id" : { "$oid" : "5d0a10d8f578c3657b340b20" }, "ArztID" : 100, "aNachname" : "Leonel", "aVorname" : "Polaschek", "aFachgebiet" : "NEUROLOGE", "patients" : [] }`
 `"_id" : { "$oid" : "5d0a1f0021c0730c9ec86931" }, "ArztID" : "1234", "aNachname" : "kul", "aVorname" : "hakan", "aFachgebiet" : "Heilmasseur", "patients" : { "PatientID" : "101", "Nachname" : "Leonardo", "Vorname" : "Dicaprio", "PLZ" : "1220", "Ort" : "wien", "Strasse" : "waehiring", "Gdat" : "05-07-1987", "ArztID" : "1234", "Diagnosse" : "krank", "ADatum" : "12-12-2019" } }`

```
mongo_client □
```

Arzt 1234 erfolgreich erstellt.

```
→ mongo_client java -jar mongo_client.jar Arzt read
```

Mit dem read Befehl kann der Arzt gesucht werden.

```
→ mongo_client
→ mongo_client java -jar mongo_client.jar Arzt update 5d0a1f0021c0730c9ec86931 NEUROLOGE
Jun 19, 2019 1:44:20 PM com.mongodb.diagnostics.logging.JULLLogger log
```

```
hek", "aFachgebiet" : "NEUROLOGE", "patients" : [ ] }
{ "_id" : { "$oid" : "5d0a1f0021c0730c9ec86931" }, "ArztID" : "1234", "aNachname" : "kul", "aVorname" : "hakan", "aFachgebiet" : "NEUROLOGE", "patients" : { "PatientID" : "101", "Nachname" : "Leonardo", "Vorname" : "Dicaprio", "PLZ" : "1220", "Ort" : "wien", "Strasse" : "waehiring", "Gdat" : "05-07-1987", "ArztID" : "1234", "Diagnosse" : "krank", "ADatum" : "12-12-2019" } }
```

Hier wird die Aktualisierung durchgeführt, Arzt mit ID 1234 wird aktualisiert.

```
terminal
→ mongo_client java -jar mongo_client.jar Arzt delete 1234
Jun 20, 2019 3:18:40 PM com.mongodb.diagnostics.logging.JULLLogger log
INFO: Cluster created with settings {hosts=[localhost:27017], mode=SINGLE, requiredClusterType=UNKNOWN}
```

Arzt wird gelöscht.

2. Use case Behandlung von MS2

In MS2 in Behandlung könnte Heilmasseur Patient ID, Heilmasseur ID Behandlungs preis, art und Datum sehen.

In MS3 habe ich die Behandlungen collection und embedded documents("Patients") & embedded documents("Praxisgemeinschaft"). Die Idee dabei ist, dass der Heilmasseur mehrere Patienten hat und auch sieht, in welcher Praxisgemeinschaft diese Patienten behandelt werden.

Behandlung

```

{
  "_id" : ObjectId("5d0b8c19dd0ac33e706b3520"),
  "HeilmasseurID" : 1,
  "patients" : [
    {
      "PatientID" : 43,
      "BhDatum" : "2015-11-24",
      "BArt" : "Heilmassage",
      "BPreis" : "88.0"
    },
    {
      "PatientID" : 96,
      "BhDatum" : "2015-04-09",
      "BArt" : "Segmentmassage",
      "BPreis" : "79.0"
    },
    {
      "PatientID" : 17,
      "BhDatum" : "2015-01-02",
      "BArt" : "Lymphdrainage",
      "BPreis" : "114.0"
    },
    {
      "PatientID" : 1,
      "BhDatum" : "2016-09-02",
      "BArt" : "Musicalergie-2",
      "BPreis" : "40.0"
    },
    {
      "PatientID" : 1,
      "BhDatum" : "2016-09-03",
      "BArt" : "Musicalergie-3",
      "BPreis" : "50.0"
    }
  ],
  "praxisgemeinschaft" : [
    {
      "PgID" : 1,
      "prName" : "HeilmassageRaum",
      "Raumnummer" : "2"
    }
  ]
}

```

```

{ "_id" : { "$oid" : "5d0a10d8f578c3657b340b20" }, "ArztID" : 100, "aNachname" : "Leonel", "aVorname" : "Polasc
hek", "aFachgebiet" : "NEUROLOGE", "patients" : [ ] }
+ mongo_client java -jar mongo_client.jar Behandlung read
Jun 19 2019 1:51:26 PM com.mongodb.diagnostics.logging.JULLogger log

```

Es wird überprüft, ob Datensätze in DB eingetragen sind.

```
1:tornike@tornike-ThinkPad-E470:~/Desktop/MS3_Khachidze_01469313/mongo_client ▼
{
  "_id": { "$oid": "5d0a10d8f578c3657b340b21" },
  "HeilmasseurID": 1,
  "patients": [
    { "PatientID": 43, "BhDatum": "2015-11-24", "BArt": "Heilmassage", "BPreis": "88.0" },
    { "PatientID": 96, "BhDatum": "2015-04-09", "BArt": "Segmentmassage", "BPreis": "79.0" },
    { "PatientID": 17, "BhDatum": "2015-01-02", "BArt": "Lymphdrainage", "BPreis": "114.0" },
    { "PatientID": 1, "BhDatum": "2016-09-02", "BArt": "Musicalergie-2", "BPreis": "40.0" },
    { "PatientID": 1, "BhDatum": "2016-09-03", "BArt": "Musicalergie-3", "BPreis": "50.0" }
  ],
  "praxisgemeinschaft": [
    { "PgID": 1, "prName": "HeilmassageRaum", "Raumnummer": "2" }
  ]
},
{
  "_id": { "$oid": "5d0a10d8f578c3657b340b22" },
  "HeilmasseurID": 2,
  "patients": [
    { "PatientID": 44, "BhDatum": "2016-09-02", "BArt": "Heilmassage", "BPreis": "118.0" }
  ]
},
{
  "_id": { "$oid": "5d0a10d8f578c3657b340b23" },
  "HeilmasseurID": 3,
  "patients": [
    { "PatientID": 56, "BhDatum": "2015-07-05", "BArt": "Lymphdrainage", "BPreis": "101.0" }
  ]
},
{
  "_id": { "$oid": "5d0a10d8f578c3657b340b24" },
  "HeilmasseurID": 4,
  "patients": []
},
{
  "_id": { "$oid": "5d0a10d8f578c3657b340b25" },
  "HeilmasseurID": 5,
  "patients": [
    { "PatientID": 36, "BhDatum": "2016-10-25", "BArt": "Moorpackung", "BPreis": "44.0" }
  ]
},
{
  "_id": { "$oid": "5d0a10d8f578c3657b340b26" },
  "HeilmasseurID": 6,
  "patients": []
},
{
  "_id": { "$oid": "5d0a10d8f578c3657b340b27" },
  "HeilmasseurID": 7,
  "patients": [
    { "PatientID": 17, "BhDatum": "2015-01-02", "BArt": "Lymphdrainage", "BPreis": "114.0" },
    { "PatientID": 8, "BhDatum": "2016-03-10", "BArt": "Segmentmassage", "BPreis": "104.0" },
    { "PatientID": 37, "BhDatum": "2016-01-20", "BArt": "Lymphdrainage", "BPreis": "72.0" },
    { "PatientID": 27, "BhDatum": "2015-07-24", "BArt": "Heilmassage", "BPreis": "51.0" }
  ]
},
{
  "_id": { "$oid": "5d0a10d8f578c3657b340b28" },
  "HeilmasseurID": 8,
  "patients": []
},
{
  "_id": { "$oid": "5d0a10d8f578c3657b340b29" },
  "HeilmasseurID": 9,
  "patients": [
    { "PatientID": 4, "BhDatum": "2015-09-22", "BArt": "Segmentmassage", "BPreis": "62.0" }
  ]
},
{
  "_id": { "$oid": "5d0a10d8f578c3657b340b2a" },
  "HeilmasseurID": 10,
  "patients": []
}

```

Alle Datensätze sind in DB.

Create Behandlung

```
Exception in thread "main" java.lang.IllegalArgumentException: no such usecase
  at Mongo_Client.main(Mongo_Client.java:30)
→ mongo_client java -jar mongo_client.jar Behandlung create 17 102 05-07-1989 Fusmassage 80 1 HeilmassageRaum
  4
Jun 19, 2019 2:05:06 PM com.mongodb.diagnostics.logging.JULLoader log
INFO: Cluster created with settings [hosts=[localhost:27017], mode=SINGLE, requiredClusterType=UNKNOWN, server
  → mongo_client
```

```

phdrainage", "BPreis": "114.0" },
  { "PatientID": 1, "BhDatum": "2016-09-02", "BArt": "Musicalergie-2", "BPreis": "40.0" },
  { "PatientID": 1, "BhDatum": "2016-09-03", "BArt": "Musicalergie-3", "BPreis": "50.0" }
],
  "praxisgemeinschaft": [
    { "PgID": 1, "prName": "HeilmassageRaum", "Raumnummer": "2" }
  ]
},
{
  "_id": { "$oid": "5d0a10d8f578c3657b340b22" },
  "HeilmasseurID": 2,
  "patients": [
    { "PatientID": 44, "BhDatum": "2016-09-02", "BArt": "Heilmassage", "BPreis": "118.0" }
  ]
},
{
  "_id": { "$oid": "5d0a10d8f578c3657b340b23" },
  "HeilmasseurID": 3,
  "patients": [
    { "PatientID": 56, "BhDatum": "2015-07-05", "BArt": "Lymphdrainage", "BPreis": "101.0" }
  ]
},
{
  "_id": { "$oid": "5d0a10d8f578c3657b340b24" },
  "HeilmasseurID": 4,
  "patients": []
},
{
  "_id": { "$oid": "5d0a10d8f578c3657b340b25" },
  "HeilmasseurID": 5,
  "patients": [
    { "PatientID": 36, "BhDatum": "2016-10-25", "BArt": "Moorpackung", "BPreis": "44.0" }
  ]
},
{
  "_id": { "$oid": "5d0a10d8f578c3657b340b26" },
  "HeilmasseurID": 6,
  "patients": []
},
{
  "_id": { "$oid": "5d0a10d8f578c3657b340b27" },
  "HeilmasseurID": 7,
  "patients": [
    { "PatientID": 17, "BhDatum": "2015-01-02", "BArt": "Lymphdrainage", "BPreis": "114.0" },
    { "PatientID": 8, "BhDatum": "2016-03-10", "BArt": "Segmentmassage", "BPreis": "104.0" },
    { "PatientID": 37, "BhDatum": "2016-01-20", "BArt": "Lymphdrainage", "BPreis": "72.0" },
    { "PatientID": 27, "BhDatum": "2015-07-24", "BArt": "Heilmassage", "BPreis": "51.0" }
  ]
},
{
  "_id": { "$oid": "5d0a10d8f578c3657b340b28" },
  "HeilmasseurID": 8,
  "patients": []
},
{
  "_id": { "$oid": "5d0a10d8f578c3657b340b29" },
  "HeilmasseurID": 9,
  "patients": [
    { "PatientID": 4, "BhDatum": "2015-09-22", "BArt": "Segmentmassage", "BPreis": "62.0" }
  ]
},
{
  "_id": { "$oid": "5d0a10d8f578c3657b340b2a" },
  "HeilmasseurID": 10,
  "patients": []
}

```

Heilmasseur mit ID 17 wurde erstellt.

```
ame" : "HeilmassageRaum", "Raumnummer" : "4" } } }
+ mongo_client java -jar mongo_client.jar Behandlung update 5d0a24f2d754211d2ef81fd7 Ruckmassage
Jun 19, 2019 2:15:14 PM com.mongodb.diagnostics.logging.JULLoader log
```

Hier wird Aktualisierung durchgeführt, Heilmasseur mit ID 17 wird aktualisiert. Bei Patient wurde Behandlungsart (Fussmassage) → durch (Ruckmassage) ersetzt.

```
_id" : { "$oid" : "5d0a10d8f578c3657b340b2a" }, "HeilmasseurID" : 10, "patients" : [ ]
_id" : { "$oid" : "5d0a24f2d754211d2ef81fd7" }, "HeilmasseurID" : "17", "patients" : { "PatientID" : "102",
shDatum" : "05-07-1989", "BArt" : "Ruckmassage", "BPreis" : "80", "praxisgemeinschaft" : { "PgID" : "1", "prN
e" : "HeilmassageRaum", "Raumnummer" : "4" } } }
mongo client
```

Delete Heilmasseur

```
ame" : "HeilmassageRaum", "Raumnummer" : "4" } } }
+ mongo_client java -jar mongo_client.jar Behandlung delete 17
Jun 19, 2019 2:16:03 PM com.mongodb.diagnostics.logging.JULLoader log
```