


Übungsprotokoll

INSY - Informationssysteme

	Übungsdatum: KW 04/2021 – KW 13/2021	Klasse: 3AHIT	Name: Felix Schneider
	Abgabedatum: 31.03.2022	Gruppe: INSY_2	Note:
Leitung: DI (FH) Alexander MESTL	Mitübende: -		
Übungsbezeichnung: MongoDB			

Inhaltsverzeichnis:

1	Aufgabenstellung.....	3
2	Abstract (English).....	3
3	Theoretische Grundlagen.....	3
4	Übungsdurchführung	3
4.1	Installation.....	3
4.2	Service verwalten	4
4.3	Nutzung	4
4.4	Überprüfung	4
4.5	Robomongo	4
4.6	Arbeiten mit MongoDB (Teil 1) – Modellierung.....	7
4.7	Arbeiten mit MongoDB (Teil 2) – Abfragen.....	27
5	Ergebnisse.....	37
6	Kommentar.....	38

1 Aufgabenstellung

Protokolliere den Ablauf der Installation und die Verwendungsmethoden von MongoDB.

2 Abstract (English)

Log the installation process and usage methods of MongoDB.

3 Theoretische Grundlagen

Bei Linux gibt es einen Spiegelserver, dem wir sagen müssen, dass wir MongoDB hinzufügen wollen.

NoSQL-Grundlagen, MongoDB-Grundlagen: https://htlkrems3500-my.sharepoint.com/personal/f_schneider_htlkrems_at/Documents/Schule/#Scripts/INSY_NoSQL.pdf
(NoSQL-Script INSY)

Die wichtigsten Begriffe in Verbindung mit NoSQL: [wichtige Begriffe INSY](#)

4 Übungsdurchführung

4.1 Installation

Folgen Sie den Anweisungen von <https://docs.mongodb.com/manual/tutorial/install-mongodb-on-debian/#import-the-public-key-used-by-the-package-management-system>, wenn Sie MongoDB auf Debian installieren wollen. Andernfalls gibt es auf dieser Website auch die Installationsdokumentationen anderer Betriebssysteme.

Trotzdem hier die wichtigsten Schritte:

Mit folgendem Befehl kann man sich den MongoDB public GPG Key importieren.

```
wget -q0 - https://www.mongodb.org/static/pgp/server-5.0.asc |  
sudo apt-key add -
```

Mit dem Befehl können Sie eine File List erstellen:

```
echo "deb http://repo.mongodb.org/apt/debian buster/mongodb-  
org/5.0 main" | sudo tee /etc/apt/sources.list.d/mongodb-org-  
5.0.list
```

Laden Sie die Pakete vom Spiegelserver neu:

```
sudo apt-get update
```

Und installieren Sie anschließend MongoDB:

```
sudo apt-get install -y mongodb-org
```

4.2 Service verwalten

Um MongoDB zu starten:

```
sudo systemctl start mongod // Dienst starten
```

Diese Befehle können Sie auch durchführen:

```
sudo systemctl status mongod // Status ansehen  
sudo systemctl enable mongod // Dienst enabeln  
sudo systemctl stop mongod // Dienst stoppen  
sudo systemctl restart mongod // Dienst neu starten
```

4.3 Nutzung

Um über das Terminal einsteigen zu können, nutzen Sie folgenden Befehl:

```
mongosh
```

4.4 Überprüfung

Mit diesem Befehl kann man nachsehen, ob ein Dienst läuft:

```
sudo systemctl status mongod
```

Mithilfe dieses Befehls kann man nachsehen, ob die Datenbank läuft (Waiting for connections):

```
tail -f /var/log/mongodb/mongod.log
```

4.5 Robomongo

4.5.1 Robomongo installieren

Gehen Sie auf folgende Website und installieren Sie die gratis Version Robo 3T:

<https://robomongo.org/download>. Entzippen Sie die Datei.

4.5.2 Robomongo ausführen

Öffnen Sie die exe-Datei im /bin-Ordner im ungezippten Ordner.

4.5.3 Connection erstellen

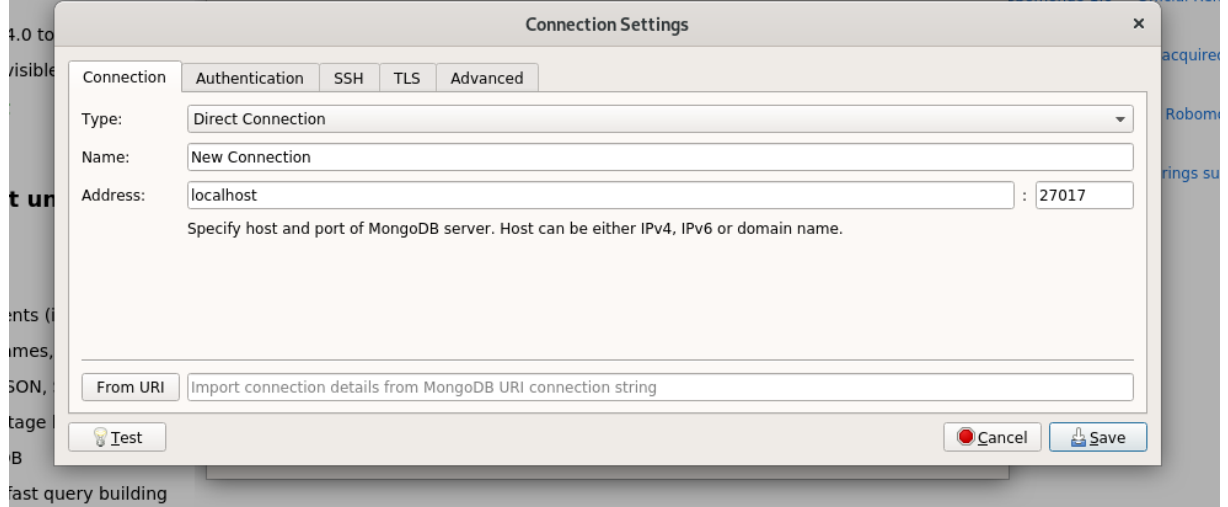
Erstellen Sie eine neue Connection mit den Standardwerte (sollte richtige Connection sein):

Features and other fixes and improvements in the latest Robo 3T release:

Robomongo is now Robo 3T, w

Wed, 14 Jun 2017

Robomongo 1.0 — Official Rel



4.5.4 Robo3T nutzen

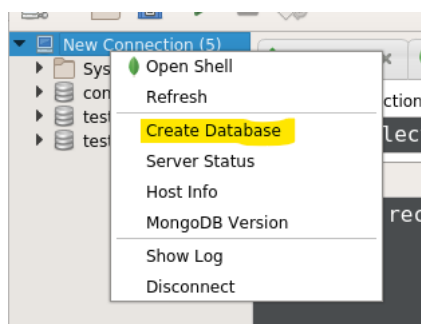
4.5.4.1 neue Database anlegen und Collection hinzufügen

Dies kann mithilfe der Shell hinzufügen:

```
test> use test3;
switched to db test3
test3> db.createCollection("test3")
{ ok: 1 }
test3> █
```

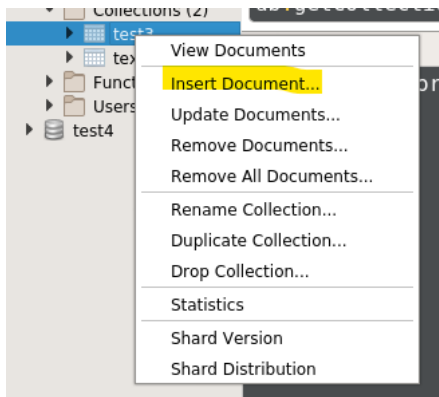
`db.createCollection(„test3“)`

Oder in Robo3T:



4.5.4.2 insertOne()

```
{ ok: 1 }
test3> db.test3.insertOne({name:"Felix"});
{
  acknowledged: true,
  insertedId: ObjectId("62173249970f1a2a5c010b9c")
}
test3> █
```



key	value	type
▼ (1) ObjectId("621732b57f65...")	{ 2 fields }	Object
_id	ObjectId("621732b57f6548b094886...")	ObjectId
klasse	3AHIT	String
▼ (2) ObjectId("621732cf970f1...")	{ 2 fields }	Object
_id	ObjectId("621732cf970f1a2a5c010...")	ObjectId
name	Felix	String

4.5.4.3 load()

```
project-data> load("/home/felix/facilities.js")
true
project-data> █
```

4.5.4.4 drop()

```
project-data> db.projects.drop();
true
. . .
```

4.5.4.5 Probleme mit NumberLong

Seit geraumer Zeit muss man bei NumberLong-Datentypen Anführungszeichen vor und nach der Zahl setzen, damit diese als String erkannt wird.

```
[,
  fundings : [{
    _id : ObjectId("874632936283294250329326"),
    debtorName : "SAP Microsystems inc.",
    amount : NumberLong(100000)
  }]
],
fundings : [{
  _id : ObjectId("874632936283294250329326"),
  debtorName : "SAP Microsystems inc.",
  amount : NumberLong("100000")
}]
```

4.5.4.6 Validation Error

Validation Errors werden nicht gut beschrieben, die Fehlermeldung ist eher mangelhaft... Trotzdem haben wir den Fehler gefunden. Die Max Length wurde überschritten, deswegen haben wir diese bei der Validation auf 100 gesetzt.

```

{
  subprojects : {
    bsonType : "array",
    items : {
      bsonType : "object",
      required : ["_id","title"],
      additionalProperties : false,
      properties : {
        _id : {
          bsonType : "objectId"
        },
        title : {
          bsonType : "string",
          minLength : 3,
          maxLength : 100,
          description : "the description of the subproject"
        }
      }
    }
  }
},

```

Nun endlich laden alle 5 Objekte:

Key	Value	Type
(1) ObjectId("5c2dfad77d1c229482d0dd9a")	{ 12 fields }	Object
(2) ObjectId("5c2dfad77d1c229482d0dd9b")	{ 12 fields }	Object
(3) ObjectId("5c2dfad77d1c229482d0dd9c")	{ 12 fields }	Object
(4) ObjectId("5c2dfad77d1c229482d0dd9d")	{ 12 fields }	Object
(5) ObjectId("5c2dfad77d1c229482d0dd9e")	{ 12 fields }	Object

4.6 Arbeiten mit MongoDB (Teil 1) – Modellierung

[Angabe wird laufend angepasst/aktualisiert - sowohl Inhalt als auch Abgabetermin!]

In einer neuen virtuellen Maschine (vorzugsweise Ubuntu 20.04. LTS) wird MongoDB in der Version 5 installiert, der Zugriff erfolgt über die grafische Oberfläche Robo3T (von Robomongo). In weiterer Folge werden Musterdaten importiert und damit gearbeitet.

Die einzelnen Steps:

- Installation Betriebssystem und MongoDB - bis 03.02.
- Installation Robo3T und Import Musterdaten - bis 17.02.
- DDL- und DML-Aufgaben aus Dokument InsyExercise, Aufgabe 4.1 - 1. bis 3. Beispiel

4.6.1 Modellierung (Schema)

4.6.1.1 Angabe

JS-Dateien (debitor und subproject) erstellen und modellieren.

4.6.1.2 Theorie

➔ NoSQL-Script S. 152ff

4.6.1.3 project.js

```
//-----  
//      schema  
//-----  
-----  
  
db.createCollection("projects", {  
  validationLevel : "strict",  
  validationAction : "error",  
  validator : {  
    $jsonSchema : {  
      bsonType : "object",  
      required : [  
        "title",  
        "projectType",  
        "projectState",  
        "description",  
        "projectBegin",  
        "isFWFSponsered",  
        "isFFGSponsered",  
        "isEUSponsered",  
        "isSmallProject",  
        "subprojects",  
        "fundings"  
      ],  
      additionalProperties : true,  
      properties : {  
        title : {  
          bsonType : "string",  
          minLength : 3,  
          maxLength : 50,  
          description : "the title of the project"  
        },  
        projectType : {  
          enum : [  
            "REQUEST_FUNDING_PROJECT",  
            "RESEARCH_FUNDING_PROJECT",  
            "MANAGEMENT_PROJECT"  
          ],  
          description : "enum to describe projecttypes"  
        },  
      },  
    },  
  },  
}
```



```

projectState : {
  enum : [
    "CREATED",
    "IN_APPROVEMENT",
    "APPROVED"
  ],
  description : "enum to describe the project state"
},
description : {
  bsonType : "string",
  minLength : 0,
  maxLength : 4000,
  description : "a short description of the
project"
},
projectBegin : {
  bsonType : "date"
},
isFWFSponsered : {
  bsonType : "bool",
  description : "indicates if the project is sponsered
by the fwf"
},
isFFGSponsered : {
  bsonType : "bool",
  description : "indicates if the project is sponsered
by the ffg"
},
isEUSponsered : {
  bsonType : "bool",
  description : "indicates if the project is sponsered
by the eu"
},
isSmallProject : {
  bsonType : "bool",
  description : "indicates that the funding of the
project don't exceeds 5000 Euro"
},
subprojects : {
  bsonType : "array",
  items : {
    bsonType : "object",
    required : ["_id","title"],
    additionalProperties : false,
    properties : {
      _id : {
        bsonType : "objectId"
      },
      title : {

```

```

        bsonType : "string",
        minLength : 3,
        maxLength : 100,
        description : "the description of the
subproject"
    }
}
},
fundings : {
    bsonType : "array",
    items : {
        bsonType : "object",
        required : ["_id", "debitorName", "amount"],
        additionalProperties : false,
        properties : {
            _id : {
                bsonType : "objectId"
            },
            debitorName : {
                bsonType : "string",
                minLength : 5,
                maxLength : 100
            },
            amount : {
                bsonType : "long",
                minimum : 0
            }
        }
    }
}
}
}
}
});

```

4.6.1.4 facilities.js

```

db.createCollection("facilities", {
    validationLevel : "strict",
    validationAction : "error",
    validator : {
        $jsonSchema : {
            bsonType : "object",
            required : [
                "_id",
                "name",
                "code",
                "projects"
            ]
        }
    }
});

```

```

    ],
    properties : {
      _id : {
        bsonType : "objectId"
      },
      name : {
        bsonType : "string",
        minLength : 3,
        maxLength : 100
      },
      code : {
        bsonType : "string",
      },
      projects : {
        bsonType : "array",
        items : {
          bsonType : "object",
          required : ["project_id", "title"],
          properties : {
            project_id : {
              bsonType : "objectId"
            },
            title : {
              bsonType : "string"
            }
          }
        }
      }
    }
  }
});

```

4.6.1.5 subproject.js

```

//-----
//      schema
//-----
-----

db.createCollection("subprojects", {
  validationLevel : "strict",
  validationAction : "error",
  validator : {
    $jsonSchema : {
      bsonType : "object",
      required : [

```

```

        "title",
        "description",
        "appliedResearch",
        "theoreticalResearch",
        "focusResearch",
        "project",
        "facilities"
    ],
    additionalProperties : true,
    properties : {
        title : {
            bsonType : "string",
            minLength : 3,
            maxLength : 100,
            description : "the title of the subproject"
        },
        description : {
            bsonType : "string",
            minLength : 0,
            maxLength : 4000,
            description : "a short description of the
subproject"
        },
        appliedResearch : {
            bsonType : "int",
            minimum : 0,
            maximum : 100
        },
        theoreticalResearch : {
            bsonType : "int",
            minimum : 0,
            maximum : 100
        },
        focusResearch : {
            bsonType : "int",
            minimum : 0,
            maximum : 100
        },
        project : {
            bsonType : "object",
            required : [ "_id", "title" ],
            additionalProperties: false,

            properties: {
                _id: {
                    bsonType: "objectId"
                },
                title: {
                    bsonType: "string",

```

```

        minLength: 3,
        maxLength: 100
      }
    },
    facilities : {
      bsonType : "object",
      required : [ "_id", "name" ],
      additionalProperties: false,

      properties: {
        _id: {
          bsonType: "objectId"
        },
        name: {
          bsonType: "string",
          minLength: 3,
          maxLength: 100
        }
      }
    }
  }
}
});

```

4.6.1.6 debtor.js

```

//-----
//      schema
//-----
-----

db.createCollection("debtors", {
  validationLevel : "strict",
  validationAction : "error",
  validator : {
    $jsonSchema : {
      bsonType : "object",
      required : [
        "name",
        "description",
        "fundings"
      ],
      additionalProperties : true,
      properties : {
        name : {
          bsonType : "string",

```

```

        minLength : 3,
        maxLength : 100,
        description : "the title of the subproject"
    },
    description : {
        bsonType : "string",
        minLength : 0,
        maxLength : 4000,
        description : "a short description of the
subproject"
    },
    fundings : {
        bsonType : "array",
        items : {
            bsonType : "object",
            required : ["_id", "title", "amount", "date"],
            additionalProperties : false,
            properties : {
                _id : {
                    bsonType : "objectId"
                },
                title : {
                    bsonType : "string",
                    minLength : 3,
                    maxLength : 50
                },
                amount : {
                    bsonType : "long",
                    minimum : 0
                },
                date : {
                    bsonType : "date"
                }
            }
        }
    }
}
});

```

4.6.2 Dokumente einfügen

4.6.2.1 Angabe

Für jede JS-Datei 5 Dokumente erstellen.

4.6.2.2 Theorie

➔ NoSQL-Script S. 170f

4.6.2.3 project.js

```
//-----  
-----  
//      insert statements  
//-----  
-----  
  
db.projects.insertMany([  
  //1. Project  
  {  
    _id : ObjectId("5c2dfad77d1c229482d0dd9a"),  
    title :      "Project 1",  
    projectType : "REQUEST_FUNDING_PROJECT",  
    projectState : "APPROVED",  
    description : "Projekt zur Planung von Produktionsplanungssystemen",  
    projectBegin : new Date ('Jun 23, 2012'),  
    isFWFSponsered : true,  
    isFFGSponsered : false,  
    isEUSponsered : false,  
    isSmallProject : false,  
    subprojects : [  
      {  
        _id : ObjectId("874632936283294250329326"),  
        title: "Subproject 3"  
      },{  
        _id : ObjectId("874632936283294250329327"),  
        title: "Subproject 4"  
      }  
    ],  
    fundings : [  
      {  
        _id : ObjectId("974632936283294250329328"),  
        debtorName : "Debitor 3",  
        amount : NumberLong(600000)  
      },  
      {  
        _id : ObjectId("974632936283294250329330"),  
        debtorName : "Debitor 5",  
        amount : NumberLong(30000)  
      }  
    ]  
  }  
]  
//2. Project  
,{  
  _id : ObjectId("5c2dfad77d1c229482d0dd9b"),  
  title :      "Project 2",  
  projectType : "REQUEST_FUNDING_PROJECT",
```

```
projectState : "APPROVED",
description  : "Projekt zur Planung von Produktionsplanungssystemen",
projectBegin : new Date ('Jun 23, 2012'),
isFWFSponsered : true,
isFFGSponsered : false,
isEUSponsered  : false,
isSmallProject : false,
subprojects : [
  {
    _id : ObjectId("874632936283294250329328"),
    title: "Subproject 5"
  },{
    _id : ObjectId("874632936283294250329332"),
    title: "Subproject 9"
  }
],
fundings : [
  {
    _id : ObjectId("974632936283294250329329"),
    debtorName : "Debitor 4",
    amount : NumberLong(150000)
  },
  {
    _id : ObjectId("974632936283294250329326"),
    debtorName : "Debitor 1",
    amount : NumberLong(120000)
  }
]
}
//3. Project
,{
  _id : ObjectId("5c2dfad77d1c229482d0dd9c"),
  title : "Project 3",
  projectType : "REQUEST_FUNDING_PROJECT",
  projectState : "APPROVED",
  description : "Projekt zur Planung von Produktionsplanungssystemen",
  projectBegin : new Date ('Jun 23, 2012'),
  isFWFSponsered : true,
  isFFGSponsered : false,
  isEUSponsered : false,
  isSmallProject : false,
  subprojects : [
    {
      _id : ObjectId("874632936283294250329324"),
      title: "Subproject 1"
    },{
      _id : ObjectId("874632936283294250329331"),
      title: "Subproject 8"
    }
  ]
}
```



```

],
fundings    : [
  {
    _id : ObjectId("974632936283294250329326"),
    debtorName : "Debitor 1",
    amount : NumberLong(67000)
  },
  {
    _id : ObjectId("974632936283294250329328"),
    debtorName : "Debitor 3",
    amount : NumberLong(300000)
  },
  {
    _id : ObjectId("974632936283294250329330"),
    debtorName : "Debitor 5",
    amount : NumberLong(350000)
  }
]
}
//4. Project
,{
  _id : ObjectId("5c2dfad77d1c229482d0dd9d"),
  title : "Project 4",
  projectType : "REQUEST_FUNDING_PROJECT",
  projectState : "APPROVED",
  description : "Projekt zur Planung von Produktionsplanungssystemen",
  projectBegin : new Date ('Jun 23, 2012'),
  isFWFSponsered : true,
  isFFGSponsered : false,
  isEUSponsered : false,
  isSmallProject : false,
  subprojects : [{
    _id : ObjectId("874632936283294250329330"),
    title: "Subproject 7"
  },{
    _id : ObjectId("874632936283294250329333"),
    title: "Subproject 10"
  }
],
fundings    : [
  {
    _id : ObjectId("974632936283294250329327"),
    debtorName : "Debitor 2",
    amount : NumberLong(12000)
  },
  {
    _id : ObjectId("974632936283294250329330"),
    debtorName : "Debitor 5",
    amount : NumberLong(1000000)
  }
]
}

```

```

    }
  ]
}
//5. Project
,{
  _id : ObjectId("5c2dfad77d1c229482d0dd9e"),
  title : "Project 5",
  projectType : "REQUEST_FUNDING_PROJECT",
  projectState : "APPROVED",
  description : "Projekt zur Planung von Produktionsplanungssystemen",
  projectBegin : new Date ('Jun 23, 2012'),
  isFWFSponsered : true,
  isFFGSponsered : false,
  isEUSponsered : false,
  isSmallProject : false,
  subprojects : [{
    _id : ObjectId("874632936283294250329325"),
    title: "Subproject 2"
  },{
    _id : ObjectId("874632936283294250329329"),
    title: "Subproject 6"
  }
  ],
  fundings : [
    {
      _id : ObjectId("974632936283294250329326"),
      debtorName : "Debitor 1",
      amount : NumberLong(300000)
    },
    {
      _id : ObjectId("974632936283294250329327"),
      debtorName : "Debitor 2",
      amount : NumberLong(10000)
    },
    {
      _id : ObjectId("974632936283294250329329"),
      debtorName : "Debitor 4",
      amount : NumberLong(90000)
    }
  ]
},
]);

```

4.6.2.4 facilities.js

```

db.facilities.insertMany([
  {
    _id : ObjectId("874632936283294250329002"),
    name : "Institut für Angewandte Mathematik",

```

```

        code : "123.456.231",
        projects : [{
            project_id : ObjectId("5c2dfad77d1c229482d0dd9c"),
            title : "Project 3"
        }]
    }, {
        _id : ObjectId("874632936283294250329001"),
        name : "Institut für Wirtschaftsinformatik",
        code : "123.456.789",
        projects : [{
            project_id : ObjectId("5c2dfad77d1c229482d0dd9a"),
            title : "Project 1"
        }, {
            project_id : ObjectId("5c2dfad77d1c229482d0dd9d"),
            title : "Project 4"
        }, {
            project_id : ObjectId("5c2dfad77d1c229482d0dd9e"),
            title : "Project 5"
        }]
    }, {
        _id : ObjectId("874632936283294250329003"),
        name : "Institut für Softwareentwicklung",
        code : "123.456.789",
        projects : [{
            project_id : ObjectId("5c2dfad77d1c229482d0dd9a"),
            title : "Project 1"
        }, {
            project_id : ObjectId("5c2dfad77d1c229482d0dd9b"),
            title : "Project 2"
        }, {
            project_id : ObjectId("5c2dfad77d1c229482d0dd9c"),
            title : "Project 3"
        }]
    }, {
        _id : ObjectId("874632936283294250329004"),
        name : "Institut für Analysis",
        code : "123.321.789",
        projects : [{
            project_id : ObjectId("5c2dfad77d1c229482d0dd9b"),
            title : "Project 2"
        }, {
            project_id : ObjectId("5c2dfad77d1c229482d0dd9d"),
            title : "Project 4"
        }, {
            project_id : ObjectId("5c2dfad77d1c229482d0dd9e");
            title : "Project 5";
        }]
    }
]);

```

4.6.2.5 subproject.js

```
// InsertMany
db.subprojects.insertMany([
  {
    _id:      ObjectId("874632936283294250329324"),
    title:    "Subproject 1",
    description: "Once upon a time some dude thought he should make
this subproject...",
    appliedResearch: 92,
    theoreticalResearch: 82,
    focusResearch: 58,
    project: {
      _id:      ObjectId("5c2dfad77d1c229482d0dd9c"),
      title:    "Project 3"
    },
    facilities: {
      _id:      ObjectId("874632936283294250329002"),
      name:     "Institut für Angewandte Mathematik"
    }
  },
  {
    _id:      ObjectId("874632936283294250329325"),
    title:    "Subproject 2",
    description: "Once upon a time some guy thought he should make this
subproject...",
    appliedResearch: 19,
    theoreticalResearch: 93,
    focusResearch: 28,
    project: {
      _id:      ObjectId("5c2dfad77d1c229482d0dd9e"),
      title:    "Project 5"
    },
    facilities: {
      _id:      ObjectId("874632936283294250329001"),
      name:     "Institut für Wirtschaftsinformatik"
    }
  },
  {
    _id:      ObjectId("874632936283294250329326"),
    title:    "Subproject 3",
    description: "Once upon a time some person thought he should make
this subproject...",
    appliedResearch: 14,
    theoreticalResearch: 8,
    focusResearch: 38,
    project: {
```

```

        _id:    ObjectId("5c2dfad77d1c229482d0dd9a"),
        title:  "Project 1"
    },
    facilities: {
        _id:    ObjectId("874632936283294250329001"),
        name:    "Institut für Wirtschaftsinformatik"
    }
},
{
    _id:    ObjectId("874632936283294250329327"),
    title:    "Subproject 4",
    description:    "Once upon a time some schurke thought he should make
this subproject...",
    appliedResearch:    28,
    theoreticalResearch:    40,
    focusResearch:    29,
    project: {
        _id:    ObjectId("5c2dfad77d1c229482d0dd9a"),
        title:    "Project 1"
    },
    facilities: {
        _id:    ObjectId("874632936283294250329003"),
        name:    "Institut für Softwareentwicklung"
    }
},
{
    _id:    ObjectId("874632936283294250329328"),
    title:    "Subproject 5",
    description:    "Once upon a time some monte thought he should make
this subproject...",
    appliedResearch:    93,
    theoreticalResearch:    82,
    focusResearch:    83,
    project: {
        _id:    ObjectId("5c2dfad77d1c229482d0dd9b"),
        title:    "Project 2"
    },
    facilities: {
        _id:    ObjectId("874632936283294250329003"),
        name:    "Institut für Softwareentwicklung"
    }
},
{
    _id:    ObjectId("874632936283294250329329"),
    title:    "Subproject 6",
    description:    "Once upon a time some monte thought he should make
this subproject...",
    appliedResearch:    45,
    theoreticalResearch:    56,

```

```

        focusResearch:      87,
        project: {
          _id:      ObjectId("5c2dfad77d1c229482d0dd9e"),
          title:    "Project 5"
        },
        facilities: {
          _id:      ObjectId("874632936283294250329004"),
          name:     "Institut für Analysis"
        }
      },
      {
        _id:      ObjectId("874632936283294250329330"),
        title:     "Subproject 7",
        description: "Once upon a time some monte thought he should make
this subproject...",
        appliedResearch:      15,
        theoreticalResearch:   94,
        focusResearch:        54,
        project: {
          _id:      ObjectId("5c2dfad77d1c229482d0dd9d"),
          title:    "Project 4"
        },
        facilities: {
          _id:      ObjectId("874632936283294250329004"),
          name:     "Institut für Analysis"
        }
      },
      {
        _id:      ObjectId("874632936283294250329331"),
        title:     "Subproject 8",
        description: "Once upon a time some monte thought he should make
this subproject...",
        appliedResearch:      37,
        theoreticalResearch:   72,
        focusResearch:        82,
        project: {
          _id:      ObjectId("5c2dfad77d1c229482d0dd9c"),
          title:    "Project 3"
        },
        facilities: {
          _id:      ObjectId("874632936283294250329003"),
          name:     "Institut für Softwareentwicklung"
        }
      },
      {
        _id:      ObjectId("874632936283294250329332"),
        title:     "Subproject 9",
        description: "Once upon a time some monte thought he should make
this subproject...",

```

```

    appliedResearch: 15,
    theoreticalResearch: 22,
    focusResearch: 11,
    project: {
      _id: ObjectId("5c2dfad77d1c229482d0dd9b"),
      title: "Project 2"
    },
    facilities: {
      _id: ObjectId("874632936283294250329004"),
      name: "Institut für Analysis"
    }
  },
  {
    _id: ObjectId("874632936283294250329333"),
    title: "Subproject 10",
    description: "Once upon a time some monte thought he should make
this subproject...",
    appliedResearch: 19,
    theoreticalResearch: 75,
    focusResearch: 38,
    project: {
      _id: ObjectId("5c2dfad77d1c229482d0dd9d"),
      title: "Project 4"
    },
    facilities: {
      _id: ObjectId("874632936283294250329001"),
      name: "Institut für Wirtschaftsinformatik"
    }
  }
]
]);

```

4.6.2.6 debtor.js

```

db.debitors.insertMany([
  //1. Debitor
  {
    _id: ObjectId("974632936283294250329326"),
    name: "Debitor 1",
    description: "some debtor",
    fundings: [
      {
        _id: ObjectId("5c2dfad77d1c229482d0dd9b"),
        title: "Project 2",
        amount: NumberLong(120000),
        date: new Date('Oct 26, 2012')
      },{
        _id: ObjectId("5c2dfad77d1c229482d0dd9c"),
        title: "Project 3",
        amount: NumberLong(67000),

```

```
        date: new Date('Sep 9, 2017')
      },{
        _id: ObjectId("5c2dfad77d1c229482d0dd9e"),
        title: "Project 5",
        amount: NumberLong(300000),
        date: new Date('Aug 17, 2018')
      }
    ]
  },
  //2. Debitor
  {
    _id: ObjectId("974632936283294250329327"),
    name: "Debitor 2",
    description: "some other debtor",
    fundings: [
      {
        _id: ObjectId("5c2dfad77d1c229482d0dd9d"),
        title: "Project 4",
        amount: NumberLong(12000),
        date: new Date('Sep 4, 2010')
      },{
        _id: ObjectId("5c2dfad77d1c229482d0dd9e"),
        title: "Project 5",
        amount: NumberLong(10000),
        date: new Date('Jan 09, 2019')
      }
    ]
  },
  //3. Debitor
  {
    _id: ObjectId("974632936283294250329328"),
    name: "Debitor 3",
    description: "this debtor",
    fundings: [
      {
        _id: ObjectId("5c2dfad77d1c229482d0dd9a"),
        title: "Project 1",
        amount: NumberLong(600000),
        date: new Date('Oct 5, 2015')
      },{
        _id: ObjectId("5c2dfad77d1c229482d0dd9c"),
        title: "Project 3",
        amount: NumberLong(300000),
        date: new Date('Feb 20, 2005')
      }
    ]
  },
  //4. Debitor
  {
```



```
_id: ObjectId("974632936283294250329329"),
name: "Debitor 4",
description: "that other debtor",
fundings: [
  {
    _id: ObjectId("5c2dfad77d1c229482d0dd9b"),
    title: "Project 2",
    amount: NumberLong(150000),
    date: new Date('Feb 6, 2019')
  }, {
    _id: ObjectId("5c2dfad77d1c229482d0dd9e"),
    title: "Project 5",
    amount: NumberLong(90000),
    date: new Date('Jan 13, 2013')
  }
]
},
//5. Debitor
{
  _id: ObjectId("974632936283294250329330"),
  name: "Debitor 5",
  description: "one debtor",
  fundings: [
    {
      _id: ObjectId("5c2dfad77d1c229482d0dd9a"),
      title: "Project 1",
      amount: NumberLong(30000),
      date: new Date('Jul 04, 2022')
    }, {
      _id: ObjectId("5c2dfad77d1c229482d0dd9c"),
      title: "Project 3",
      amount: NumberLong(350000),
      date: new Date('Sep 29, 2021')
    }, {
      _id: ObjectId("5c2dfad77d1c229482d0dd9d"),
      title: "Project 4",
      amount: NumberLong(1000000),
      date: new Date('Oct 2, 2019')
    }
  ]
}
]);
```

4.6.3 Dokumente bearbeiten

4.6.3.1 Angabe

Bestimmte Attribute bzw. Felder verändern und updaten, löschen oder erstellen.

4.6.3.2 Theorie

➔ NoSQL-Script S. 171ff

4.6.3.3 update1.js

Jedem Projekt ein Enddatum geben, das Rating ersetzen und einen Partner setzen (erstellen).

```
db.projects.updateMany(
  {},
  {
    $set: {
      projectEnd: new Date('Jan 01, 2020'),
      rating: 5,
      partners: [ { name: "TU Wien" } ]
    }
  }
);
```

4.6.3.4 update2.js

Alle REQUEST_FUNDING_PROJECT – Projekte EU-gesponsert markieren.

```
db.projects.updateMany(
  {
    projectType : "REQUEST_FUNDING_PROJECT"
  },
  {
    $set: { isEUSponsered: true }
  }
);
```

4.6.3.5 update3.js

Das rating zu projectRating umbenennen, weil da ein Fehler beim Schema passiert ist.

```
db.projects.updateMany(
  {},
  {
    $rename: { rating: "projectRating" }
  }
);
```

4.6.3.6 update4.js

Zwei Partner hinzufügen und einen wieder entfernen, weil der doch nicht hinzugehört...

```
db.projects.updateMany(
  {},
  {
    $push: {
      partners: {
        $each: [
```

```
        { name: "HTL Krems" },
        { name: "TU Graz" }
      ]
    }
  }
};
db.projects.updateMany(
  {},
  {
    $pull: {
      partners: {
        name: "HTL Krems"
      }
    }
  }
);
```

4.7 Arbeiten mit MongoDB (Teil 2) – Abfragen

4.7.1 1. Aufgabe (hat 3 Unteraufgaben)

1.) Finden Sie alle project Dokumente in der projects collection.

- *) Geben Sie nur die ersten 5 Project Dokumente aus.
- *) Sortieren Sie die projecte nach dem titel aufsteigend
- *) Geben Sie fuer die project Dokumente jeweils den Titel, den Type und den Projektzustand aus.

Mithilfe des .limit(5) beschränken wir die Ausgabe auf maximal 5 Elemente.

Mithilfe des .sort({title: 1}) sortieren wir die Ausgabe aufsteigend nach dem title.

In der ersten Klammer des find-Befehls findet die Selection statt, sprich wir filtern bzw. selectieren die Collection (bei späteren Aufgaben).

In der zweiten Klammer des find-Befehls findet die Projection statt, sprich wir geben fest, welche Felder ausgegeben werden sollen.

New Connection localhost:27017 project

```
db.getCollection('projects').find({}, {title:1,projectType:1,projectState:1}).sort({title:1}).limit(5)
```

projects 0.002 sec.

Key	Value	Type
(1) ObjectId("5c2dfad77d1c229482d0dd9d")	{ 4 fields }	Object
_id	ObjectId("5c2dfad77d1c229482d0dd9d")	ObjectId
title	Finite Elemente	String
projectType	RESEARCH_FUNDING_PROJECT	String
projectState	APPROVED	String
(2) ObjectId("5c2dfad77d1c229482d0dd9b")	{ 4 fields }	Object
_id	ObjectId("5c2dfad77d1c229482d0dd9b")	ObjectId
title	Generische Programmierung	String
projectType	REQUEST_FUNDING_PROJECT	String
projectState	APPROVED	String
(3) ObjectId("5c2dfad77d1c229482d0dd9e")	{ 4 fields }	Object
(4) ObjectId("5c2dfad77d1c229482d0dd9a")	{ 4 fields }	Object
(5) ObjectId("5c2dfad77d1c229482d0dd9c")	{ 4 fields }	Object

2.) Finden Sie alle REQUEST_FUNDING_PROJECTe die sich im Zustand APPROVED befinden.

- *) Geben Sie nur die ersten 5 Project Dokumente aus.
- *) Sortieren Sie die projecte nach dem titel aufsteigend
- *) Geben Sie fuer die project Dokumente jeweils den Titel, den Type und den Projektzustand aus.

Hier benötigen wir nun Selection, Projection, .sort() und .limit()

```
db.getCollection('projects').find(
{
  projectType: "REQUEST_FUNDING_PROJECT",
  projectState: "APPROVED"
},{
  title: 1,
  projectType: 1,
  projectState: 1
}).sort({title: 1}).limit(5)
```

projects 0.001 sec.

Key	Value	Type
(1) ObjectId("5c2dfad77d1c229482d0dd9b")	{ 4 fields }	Object
_id	ObjectId("5c2dfad77d1c229482d0dd9b")	ObjectId
title	Generische Programmierung	String
projectType	REQUEST_FUNDING_PROJECT	String
projectState	APPROVED	String
(2) ObjectId("5c2dfad77d1c229482d0dd9a")	{ 4 fields }	Object
_id	ObjectId("5c2dfad77d1c229482d0dd9a")	ObjectId
title	Produktionsplanungssysteme	String
projectType	REQUEST_FUNDING_PROJECT	String
projectState	APPROVED	String

//-----

3.) Finden alle Subprojecte deren appliedResearch oder theoreticalResearch oder focusResearch einen Wert ueber 60 vorweisen.

*) Geben Sie nur die ersten 3 Subproject Dokumente aus.

*) Sortieren Sie die subprojecte nach dem titel aufsteigend

*) Geben Sie fuer die subproject Dokumente jeweils den Titel und die Forschungsschwerpunkte aus.

Mithilfe von \$or bzw. \$and kann man bei der Selection festlegen, dass nur eines bzw. alle Argumente erfüllt werden.

```
db.getCollection('subprojects').find({
  $or: [
    { appliedResearch: { $gt: 60 } },
    { theoreticalResearch: { $gt: 60 } }
  ],
  {
    title: 1,
    description: 1
  }
}).sort({title: 1}).limit(3)
```

subprojects 0.001 sec.	
key	Value
▼ (1) ObjectId("874632936283294250329324")	{ 3 fields }
_id	ObjectId("874632936283294250329324")
title	ERP Sap
description	Subproject zur Bestimmung von ERP Sap
▶ (2) ObjectId("874632936283294250329008")	{ 3 fields }

4.7.2 2. Aufgabe (hat 4 Unteraufgaben)

1.) Finden Sie alle subproject Dokumente aus die vom Institut fuer Softwareentwicklung, durchgefuehrt werden.

*) Sortieren Sie die subprojecte nach dem titel

Um auf Felder in einem Array zugreifen zu können, muss man diesen Ausdruck unter Anführungszeichen setzen („facility.name“):

```
db.getCollection('subprojects').find({
  "facility.name": "Institut für Softwareentwicklung"
},{}).sort({title: 1})
```

subprojects 0.001 sec.		
Key	Value	Type
▶ (1) ObjectId("874632936283294250329001")	{ 8 fields }	Object
▼ (2) ObjectId("874632936283294250329003")	{ 8 fields }	Object
_id	ObjectId("874632936283294250329003")	ObjectId
title	Transaktionsverfahren - MVCC	String
description	Erforschen von Transaktionsverfahren	String
project_id	ObjectId("5c2dfad77d1c229482d0dd9c")	ObjectId
appliedResearch	50	Int32
theoreticalResearch	50	Int32
focusResearch	0	Int32
facility	{ 3 fields }	Object
_id	ObjectId("874632936283294250329003")	ObjectId
name	Institut für Softwareentwicklung	String
code	123.456.789	String
▶ (3) ObjectId("874632936283294250329002")	{ 8 fields }	Object

2.) Finden Sie alle subproject Dokumente deren Forschungsschwerpunkte in Summe 100 ergeben.

*) Sortieren Sie die subprojecte nach dem titel aufsteigend.

Die \$where-Klausel: Mithilfe der \$where Klausel, kann man eine function definieren und in dieser Function JavaScript nutzen. Dies erleichtert ALLES!!!

```
db.getCollection('subprojects').find({
  $where: function() { return this.appliedResearch + this.theoreticalResearch + this.focusResearch == 100; }
},{}).sort({title: 1})
```

subprojects 0.188 sec.		
Key	Value	Type
▼ (1) ObjectId("874632936283294250329325")	{ 8 fields }	Object
_id	ObjectId("874632936283294250329325")	ObjectId
title	ERP Dynamics	String
description	Subproject zur Bestimmung von ERP Dynamics	String
project_id	ObjectId("5c2dfad77d1c229482d0dd9a")	ObjectId
appliedResearch	20	Int32
theoreticalResearch	20	Int32
focusResearch	60	Int32
facility	{ 3 fields }	Object
▶ (2) ObjectId("874632936283294250329324")	{ 8 fields }	Object
▶ (3) ObjectId("874632936283294250329006")	{ 8 fields }	Object
▶ (4) ObjectId("874632936283294250329004")	{ 8 fields }	Object
▶ (5) ObjectId("874632936283294250329005")	{ 8 fields }	Object
▼ (6) ObjectId("874632936283294250329007")	{ 8 fields }	Object
_id	ObjectId("874632936283294250329007")	ObjectId
title	Motorsimulation - Mathematische Methoden	String
description	Motorsimulation - Mathematische Methoden	String
project_id	ObjectId("5c2dfad77d1c229482d0dd9e")	ObjectId
appliedResearch	35	Int32
theoreticalResearch	25	Int32
focusResearch	40	Int32
facility	{ 3 fields }	Object
▶ (7) ObjectId("874632936283294250329008")	{ 8 fields }	Object
▶ (8) ObjectId("874632936283294250329001")	{ 8 fields }	Object
▶ (9) ObjectId("874632936283294250329003")	{ 8 fields }	Object
▶ (10) ObjectId("874632936283294250329002")	{ 8 fields }	Object

3.) Finden Sie alle debitoren die mehr als 2 Projekte finanzieren

*) Sortieren Sie die debitoren nach dem Namen

```
db.getCollection('debtors').find({
  $where: function() { return this.fundings.length >= 2; }
},{}).sort({name: 1})
```

debtors 0.214 sec.		
Key	Value	Type
▼ (1) ObjectId("874632936283294250329115")	{ 4 fields }	Object
_id	ObjectId("874632936283294250329115")	ObjectId
name	Simens Microsystems inc.	String
description	Simens Microsystems inc.	String
fundings	[2 elements]	Array
▶ [0]	{ 4 fields }	Object
▼ [1]	{ 4 fields }	Object
project_id	ObjectId("5c2dfad77d1c229482d0dd9e")	ObjectId
projectName	Motorensimulation	String
amount	3000	Int64
transferredAt	2018-10-31 23:00:00.000Z	Date
▶ (2) ObjectId("874632936283294250329114")	{ 4 fields }	Object

4.) Finden Sie alle projecte die mit mehr als 700000 finanziert sind.

*) Sortieren Sie die projecte nach dem titel.

```
db.getCollection('projects').find({
  $where: function() {
    var money = 0;
    for(var i = 0; i < this.fundings.length; i++) {
      money += this.fundings[i].amount;
    }
    return money > 700000;
  }
}, {}).sort({title: 1})
```

projects 0.004 sec.		
Key	Value	Type
(1) ObjectId("5c2dfad77d1c229482d0dd9d")	{ 12 fields }	Object
_id	ObjectId("5c2dfad77d1c229482d0dd9d")	ObjectId
title	Finite Elemente	String
projectType	RESEARCH_FUNDING_PROJECT	String
projectState	APPROVED	String
description	Projekt zur Erforschung der Methoden der Finiten Elemente in ...	String
projectBegin	2017-09-09 22:00:00.000Z	Date
isFWFSponsored	false	Boolean
isFFGSponsored	false	Boolean
isEUSponsored	true	Boolean
isSmallProject	false	Boolean
subprojects	[3 elements]	Array
fundings	[3 elements]	Array
[0]	{ 3 fields }	Object
_id	ObjectId("874632936283294250329114")	ObjectId
debitorName	Sun Microsystems inc.	String
amount	500000	Int64
[1]	{ 3 fields }	Object
_id	ObjectId("874632936283294250329115")	ObjectId
debitorName	Simens Microsystems inc.	String
amount	400000	Int64
[2]	{ 3 fields }	Object
_id	ObjectId("874632936283294250329116")	ObjectId
debitorName	TU Wien	String
amount	150000	Int64

4.7.3 3. Aufgabe (hat 3 Unteraufgaben)

1.) Finden Sie alle Projekte die weder
REQUEST_FUNDING_PROJECTS noch
RESEARCH_FUNDING_PROJECTS
sind.

*) Für die Projekte soll nur der Titel und der
Projekttyp angegeben werden.

*) Sortieren Sie das Ergebnis nach dem Titel
absteigend

Es gibt verschiedene Lösungswege (logischerweise) für diese Aufgabe. Ich habe mich dazu entschieden, false zu returnen, wenn der projectType entweder RESEARCH_FUNDING_PROJECT oder REQUEST_FUNDING_PROJECT ist.


```
db.getCollection('projects').find({
  $where: function() {
    if(this.projectType == "RESEARCH_FUNDING_PROJECT")
      return false;
    if(this.projectType == "REQUEST_FUNDING_PROJECT")
      return false;
    return true;
  }
},{
  title: 1,
  projectType: 1
}).sort({title: -1})
```

projects 0.007 sec.	
Key	Value
(1) ObjectId("5c2dfad77d1c229482d0dd9e")	{ 3 fields }
_id	ObjectId("5c2dfad77d1c229482d0dd9e")
title	Motorensimulation
projectType	MANAGEMENT_PROJECT

2.) Finden Sie alle Projekte die weder REQUEST_FUNDING_PROJECTS noch RESEARCH_FUNDING_PROJECTS sind. Die Projekte muessen 1 Subprojekt haben. Zustzlich muessen die Projekt die TU Wien als Partner haben.

- *) Fuer die Projekte soll nur der Titel und der Projekttyp angegeben werden.
- *) Sortieren Sie das Ergebnis nach dem titel absteigend

Der JS Code bei meiner Lösung hat extra Komplexität, damit es schwieriger ist, ihn zu verstehen... 😊

```
db.getCollection('projects').find({
  $where: function() {
    if(this.projectType == "RESEARCH_FUNDING_PROJECT")
      return false;
    if(this.projectType == "REQUEST_FUNDING_PROJECT")
      return false;
    if(this.subprojects.length != 1)
      return false;
    for(var i = 0; i < this.fundings.length; i++) {
      if(this.fundings[i].debitorName == "TU Wien")
        return true;
    }
    return false;
  }, {
    title: 1,
    projectType: 1
  }).sort({title: -1})
```

Mit den Updates hinzugefügt bekommt man ebenfalls keine Ergebnisse:

```
db.getCollection('projects').find({
  $where: function() {
    if(this.projectType == "RESEARCH_FUNDING_PROJECT")
      return false;
    if(this.projectType == "REQUEST_FUNDING_PROJECT")
      return false;
    if(this.subprojects.length != 1)
      return false;
    for(var i = 0; i < this.partners.length; i++) {
      if(this.partners[i].name == "TU Wien")
        return true;
    }
    return false;
  }
})
```

🕒 0.093 sec.

Fetches 0 record(s) in 93ms

3.) Finden Sie alle Projekte die ein Rating zwischen 3 und 5 haben. Die Projekte müssen als Partner sowohl die TU Wien als auch die TU Graz haben.

Der einfachste Weg, um dies zu lösen, ist wahrscheinlich, zwei Variablen zu deklarieren, und nur true zu returnen, wenn beide true sind.

```
db.getCollection('projects').find({
  $where: function() {
    if(this.rating >= 3 && this.rating <= 5)
      return false;
    var tu_wien = false;
    var tu_graz = false;
    for(var i = 0; i < this.fundings.length; i++){
      if(this.fundings[i].debitorName == "TU Wien"){
        tu_wien = true;
      }
      if(this.fundings[i].debitorName == "TU Graz"){
        tu_graz = true;
      }
    }
    return (tu_wien && tu_graz);
  }
},{
  {
```

0.105 sec.

Fetches 0 record(s) in 105ms

Mit den Updates hinzugefügt bekommt man sogar Ergebnisse:

<pre> db.getCollection('projects').find({ \$where: function() { if(this.projectRating < 3 && this.projectRating > 5) return false; var tu_wien = false; var tu_graz = false; for(var i = 0; i < this.partners.length; i++) { if(this.partners[i].name == "TU Wien") tu_wien = true; if(this.partners[i].name == "TU Graz") tu_graz = true; } return (tu_wien && tu_graz); }, { }) </pre>		
projects 0.005 sec.		
Key	Value	Type
(1) ObjectId("5c2dfad77d1c229482d0dd9a") <ul style="list-style-type: none"> _id: ObjectId("5c2dfad77d1c229482d0dd9a") title: Produktionsplanungssysteme projectType: REQUEST_FUNDING_PROJECT projectState: APPROVED description: Projekt zur Planung von Produktionsplanungssystemen projectBegin: 2012-06-22 22:00:00.000Z isFWFSponsered: true isFFGSponsered: false isEUSponsered: true isSmallProject: false subprojects: [2 elements] fundings: [1 element] partners: [2 elements] <ul style="list-style-type: none"> [0]: { 1 field } <ul style="list-style-type: none"> name: TU Wien [1]: { 1 field } <ul style="list-style-type: none"> name: TU Graz projectEnd: 2019-12-31 23:00:00.000Z projectRating: 5 		Object
(2) ObjectId("5c2dfad77d1c229482d0dd9b")	{ 15 fields }	Object
(3) ObjectId("5c2dfad77d1c229482d0dd9c")	{ 15 fields }	Object
(4) ObjectId("5c2dfad77d1c229482d0dd9d")	{ 15 fields }	Object
(5) ObjectId("5c2dfad77d1c229482d0dd9e")	{ 15 fields }	Object

6 Kommentar

Project	Debitors	Subproject	Facilities
1	3,5	3,4	1,3
2	4,1	5,9	3,4
3	1,3,5	1,8	2,3
4	2,5	7,10	4,1
5	1,2,4	2,6	1,4

Falls Sie nachvollziehen wollen, welche Projects welche Subprojects, Facilities und so haben...