SVEUČILIŠTE U ZAGREBU
FAKULTET ORGANIZACIJE I INFORMATIKE
VARAŽDIN

Antun Tkalčec

APLIKACIJA ZA ANALIZU VELIKIH KOLIČINA PODATAKA

PROJEKT

TEORIJA BAZA PODATAKA

Varaždin, 2021.

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Antun Tkalčec

Matični broj: 0016136241

Studij: Baze podataka i baze znanja

APLIKACIJA ZA ANALIZU VELIKIH KOLIČINA PODATAKA

PROJEKT

Mentor:

dr. sc. Bogdan Okreša Đurić

Varaždin, siječanj 2021.

Antun Tkalčec

Izjava o izvornosti

Izjavljujem da je moj projekt izvorni rezultat mojeg rada te da se u izradi istoga nisam koristio drugim izvorima osim onima koji su u njemu navedeni. Za izradu rada su korištene etički prikladne i prihvatljive metode i tehnike rada.

Autor potvrdio prihvaćanjem odredbi u sustavu FOI Radovi

Sažetak

Ovaj projektni rad napravljen je na temu aplikacije za analizu velikih količina podataka. Rad se bavi polustrukturiranim modelom baza podataka, točnije MongoDB-om i vršenjem upita nad bazom podataka u alatu Jupyter Notebook. Rad prvo postavlja teorijsku podlogu te motivaciju za radom sa takvim modelom baze podataka, a polazi od teze da je polustrukturirani model u današnje vrijeme često bolji model od relacijskog pri radu s velikim količinama podataka. Zaključak rada je da će, budući da količina podataka za spremanje i analizu raste eksponencijalno, alati poput MongoDB, Jupyter Notebook i programskog jezika Python biti od sve veće važnosti za struku.

Ključne riječi: NoSQL, polustrukturirane BP, MongoDB, Jupyter Notebook, model baze podataka

Sadržaj

1. Opis aplikacijske domene

Analiza velikih količina podataka proces je otkrivanja trendova, uzoraka i veza između velikih količina podataka radi boljeg donošenja odluka pomoću dobivenih podataka. Zadnjih godina primjećuje se nagli porast količine podataka, a pomoću novih tehnologija poput NoSQL baza podataka, inženjeri podataka pronalaze nove načine spremanja i procesiranja tih podataka. Postupak rada s velikom količinom podataka je sljedeći [1]:

- prikupljanje podataka
- procesiranje podataka
- čišćenje podataka
- analiziranje podataka

Takav rad s *"big data"* aplikacijska je domena ovog projektnog rada. Projekt će se baviti upravo NoSQL bazama podataka, koje pripadaju polustrukturiranim bazama podataka. Koristit će se MongoDB, a sučelje će biti implementirano u Jupyter Notebooku. Autorova motivacija za izbor ove teme i ovih tehnologija leži u činjenici da se ovakve baze podataka i navedene tehnologije sve više koriste u praksi te je osobno mišljenje autora da je znanje osnova rada sa MongoDB i sličnim tehnologijama izuzetno vrijedno u današnjem svijetu.

2. Teorijski uvod

Projekt se, dakle, bavi polustrukturiranim modelom baza podataka koji je evolucija relacijskog modela te su podaci spremljeni u fleksibilnu strukturu koje zovemo **kolekcijom**. U ovom modelu ne postoji odvajanje podataka i *scheme*, a kolekcije nisu rigidne u svojoj strukturi kao tablice kod relacijskog modela. Unutar kolekcije, nekim objektima mogu nedostajati atributi, a drugi objekti mogu imati više od ostalih [2]. Tip atributa objekta je fleksibilan - dok u relacijskom modelu atribut poprima npr. *string*, kod polustrukturiranog modela tip može biti neka druga kolekcija.

Ugniježđeni atributi su samo jedna od prednosti ovog modela nad relacijskim, gdje bismo morali postaviti vanjske ključeve na druge tablice koje bi sadržavale podatke o danom atributu. Neke od ostalih prednosti su [3]:

- lako je promijeniti strukturu
- podaci su lako prenosivi
- **lak rad sa heterogenim izvorima**

Upravo posljednja od navedenih prednosti je, po osobnom mišljenju autora, najveći razlog porasta korištenja ovog modela. Današnji inženjeri podataka kao zadatak često imaju *skrejpati* podatke iz različitih *web* izvora, koji gotovo nikad neće imati istu strukturu.

Međutim, postoje nedostaci polustrukturiranog modela u usporedbi s relacijskim [3]:

- upiti su manje efikasni - zapisi u ovom modelu su spremljeni na disku i imaju svoje pokazivače, te upiti moraju *tražiti* po disku sljedeći pokazivače
- troškovi pohrane su veći
- teža interpretacija veza između podataka

Unatoč ovim i ostalim nedostacima, učestalost korištenja ovog modela i tehnologija poput MongoDB i Pandas raste.

Primjeri polustrukturiranih tipova podataka su [4]:

- email
- .csv - podaci izraženi kao *"Ivica, Pivica, Njivica"*, gdje je *delimiter* znak zareza
- XML - *"extensible markup language"*
- JSON - *"Javascript Object Notation"*

Izgled XML-a i JSON-a je sličan. Primjer iste datoteke u XML-u i JSON-u može ovako izgledati [5]:

XML:

```
<menu id="file" value="File">
  <popup>
    <menuitem value="New" onclick="CreateNewDoc()" />
    <menuitem value="Open" onclick="OpenDoc()" />
    <menuitem value="Close" onclick="CloseDoc()" />
  </popup>
</menu>
```

Slika 1: Primjer XML zapisa podataka; preuzeto sa http://www.json.org/example.html 17.12.2021.

Isti zapis u **JSON** formatu izgleda ovako:

```
{"menu": {
  "id": "file",
  "value": "File",
  "popup": {
    "menuitem": [
      {"value": "New", "onclick": "CreateNewDoc()"},
      {"value": "Open", "onclick": "OpenDoc()"},
      {"value": "Close", "onclick": "CloseDoc()"}
    ]
  }
}}
```

Slika 2: Primjer JSON zapisa podataka; preuzeto sa http://www.json.org/example.html 17.12.2021.

3. Model baze podataka

Polustrukturirani model se, kako je već objašnjeno, ne koristi tablicama kao relacijske baze podataka, već fleksibilnim strukturama. MongoDB nudi dvije vrste modela podataka [6]:

- ugrađeni model - *denormalizirani* model, gdje se svi povezani podaci prikazuju u istom dokumentu, i
- normalizirani model - pod-dokumenti se referencama vežu za dokumente, poput vanjskih ključeva

Općenito se ove vrste modela koriste prema potrebama korisnika, a ovaj projekt će koristiti ugrađeni model. Kao bazu podataka projekt će koristiti dokument "Pivo", koji će imati atribute poput naziva, marketinškog slogana, postotka alkohola, sastojaka itd. Neki od atributa će biti ugrađeni u druge, tj. postoji više vrsta sastojaka, a unutar svake vrste sastojaka može postojati nekoliko različitih sastojaka.

Podaci, kao i struktura dokumenta, tj. kolekcija "Pivo" su preuzeti s *punkapi*-ja [7], spremljeni u JSON datoteku iz koje su određeni podaci ručno upisani u MongoDB. Konkretno, izgled JSON polja je ovakav:

Isječak kôda 1: Model baze podataka

```
[
  {
    "id": 192,
    "name": "Punk IPA 2007 - 2010",
    "tagline": "Post Modern Classic. Spiky. Tropical. Hoppy.",
    "first_brewed": "04/2007",
    "abv": 6.0,
    "ph": 4.4,
    "volume": {
      "value": 20,
      "unit": "liters"
    },
    "boil_volume": {
      "value": 25,
      "unit": "liters"
    },
    "ingredients": {
      "malt": [
        {
          "name": "Extra Pale",
          "amount": {
            "value": 5.3,
            "unit": "kilograms"
          }
        }
      ],
```

```
      "hops": [
        {
          "name": "Ahtanum",
          "amount": {
            "value": 17.5,
            "unit": "grams"
          },
          "add": "start",
          "attribute": "bitter"
        },
        {
          "name": "Chinook",
          "amount": {
            "value": 15,
            "unit": "grams"
          },
          "add": "start",
          "attribute": "bitter"
        },
        ...
      ],
      "yeast": "Wyeast 1056 - American ™Ale"
    },
  }
]
```

Gornji kod prikazuje model baze podataka te primjer vrijednosti za pivo koje može biti uneseno u MongoDB. Kako je već spomenuto, ovakav model podataka može imati ugniježđene atribute, što se može vidjeti kod npr. *volume*, koji sadrži svoja dva unutarnja atributa *value* i *unit*. U relacijskom modelu baze podataka bi za tako nešto trebali napraviti novu tablicu i povezati ih vanjskim ključem, no ovdje se podaci spremaju u jedno polje, koje je programskim jezicima poput PHP-a jednostavno *decode*-ati i programski pohraniti u bazu podataka. Ipak, za potrebe ovog projekta, podaci će biti ručno uneseni za manji broj piva.

4. Implementacija

U ovom poglavlju bit će prikazan rad u terminalu, gdje je napravljena veza sa MongoDB-jem te izrađivani upiti za upisivanje podataka. Također, bit će prikazan i način rada u Jupyter Notebooku.

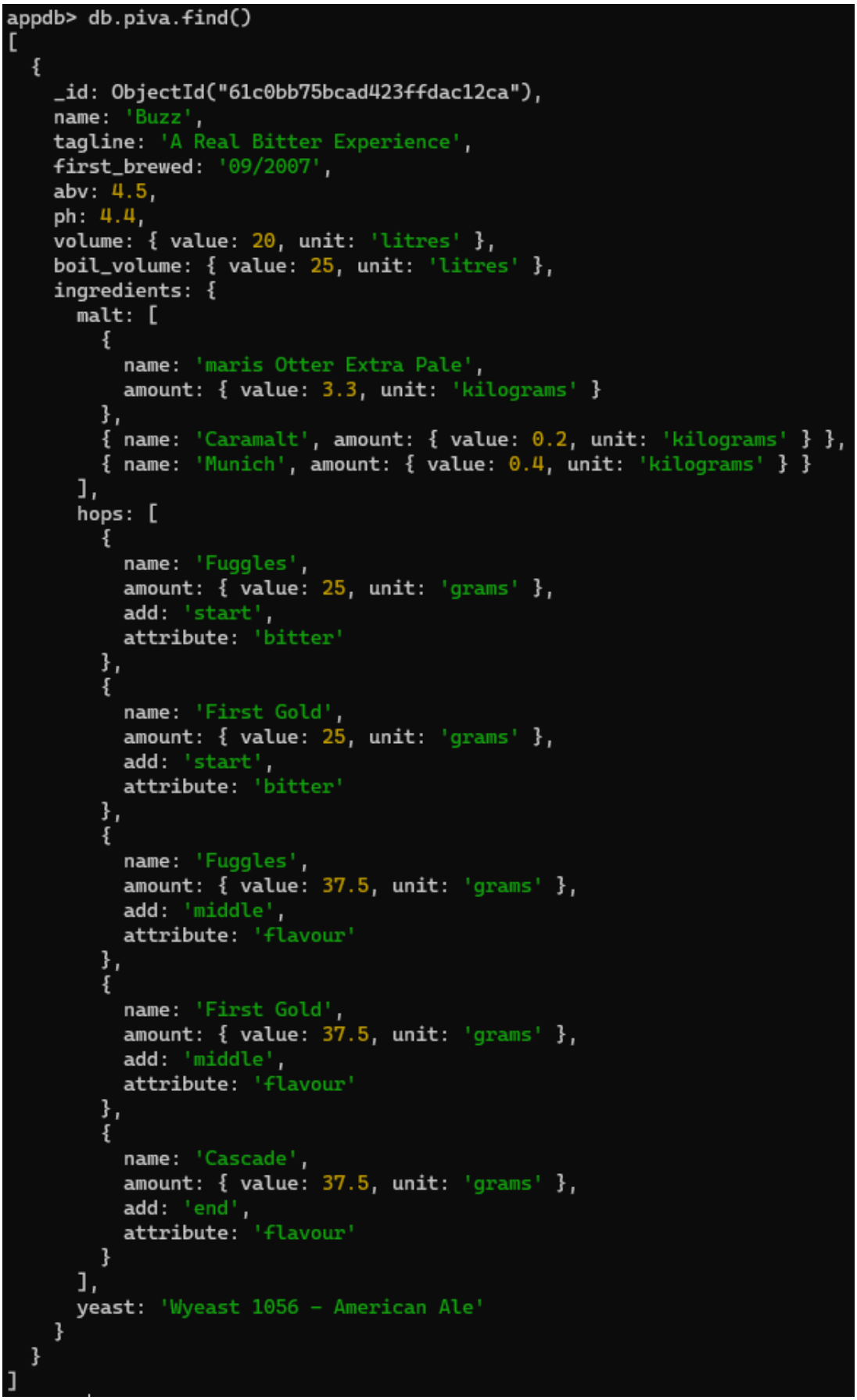
4.1. Rad u terminalu

Veza s MongoDB je stvorena unutar *Windows Terminal*. Stvara se komandom *mongosh.exe*, a pozicioniranje u proizvoljnu bazu podataka vrši se sa *use <ime BP>*. Baza ovog projekta naziva se *appdb*, iz čega proizlazi *use appdb*. MongoDB funkcionira na način da se baza podataka stvara tek kada u njoj nešto postoji, stoga da bi se baza podataka *appdb* pojavila nakon komande *show dbs*, potrebno je u bazu upisati podatke.

Rad s MongoDB odmah daje sliku rada s *NoSQL* bazom podataka. Pohrana podataka se ne vrši sa *INSERT INTO ...* upitom, već sa funkcijom koja sintaksom sliči *JavaScriptu*, "db.piva.insertOne(...)". Analogno, za upis više podataka odjednom koristi se "db.piva.insertMany(...)", no zbog kompleksnosti podataka koji se upisuju u bazu u ovom slučaju, koristit će se *insertOne* verzija.

Slika 3: *Screenshot* terminala s upisom podataka; autorov uradak

Nakon što se ovako upiše pivo u kolekciju, ono se može pronaći sa "db.piva.find()", što je funkcija za vraćanje svih zapisa unutar kolekcije piva.

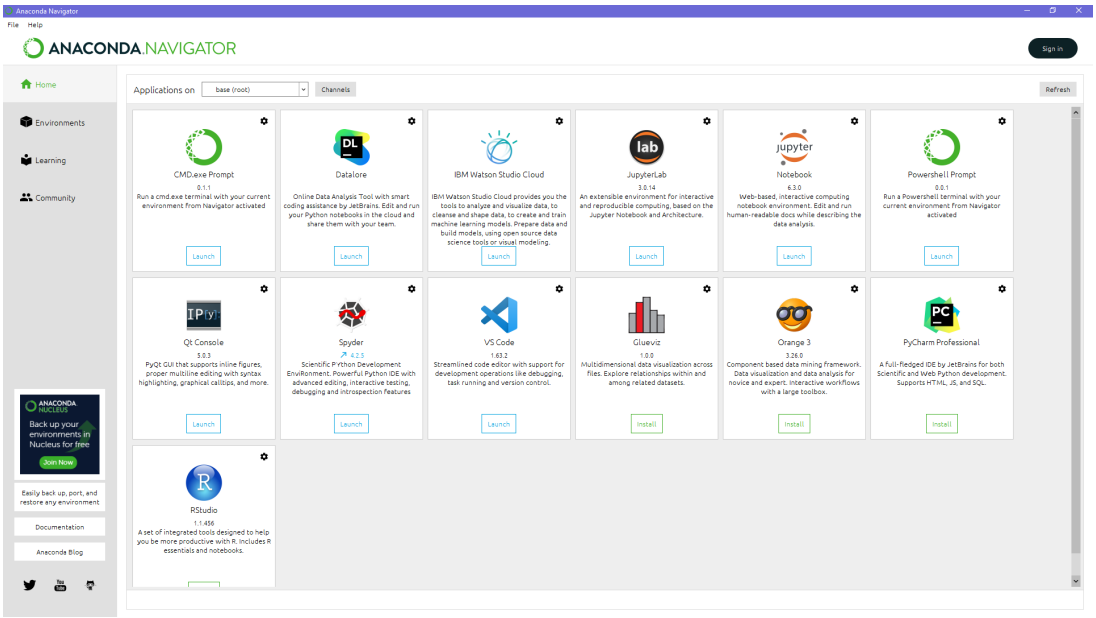


Slika 4: *Screenshot* terminala u kojem je pronađeno prethodno upisano pivo; autorov uradak

Na isti način su upisana još dva piva, koja se mogu na isti način pretraživati.

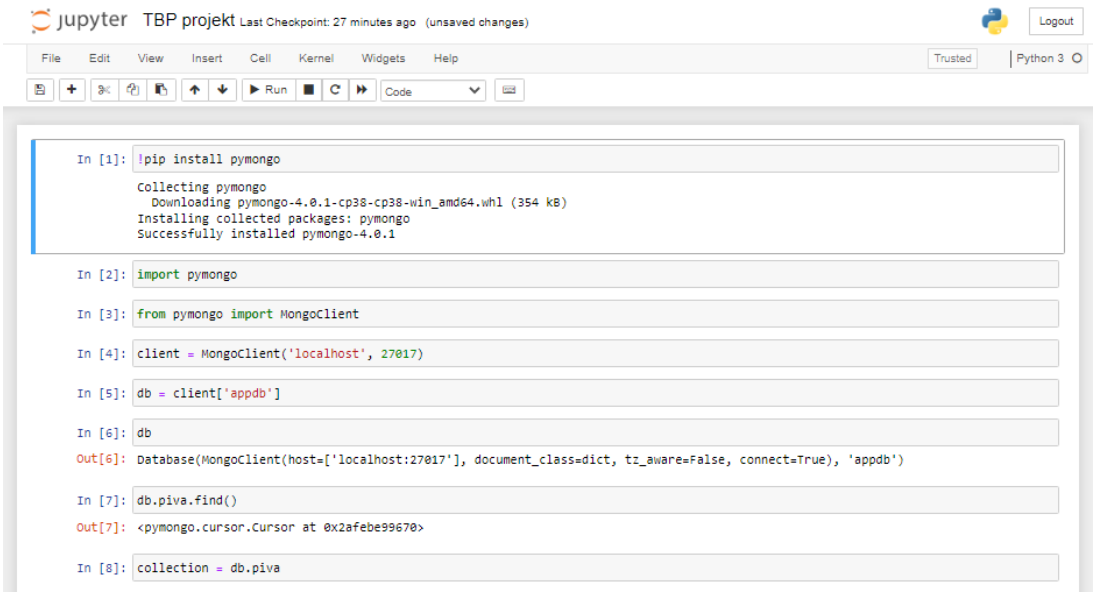
4.2. Jupyter Notebook

Jupyter Notebook je *web* aplikacija koju inženjeri podataka koriste za analizu velikih količina podataka koristeći programski jezik Python i njegove biblioteke poput Pandas i pymongo. Na autorovom računalu je već prije izrade ovog projekta instalirana platforma Anaconda, koja dolazi sa već instaliranim Jupyter Notebookom. Notebook se pali tako da se pokrene Anaconda Navigator.



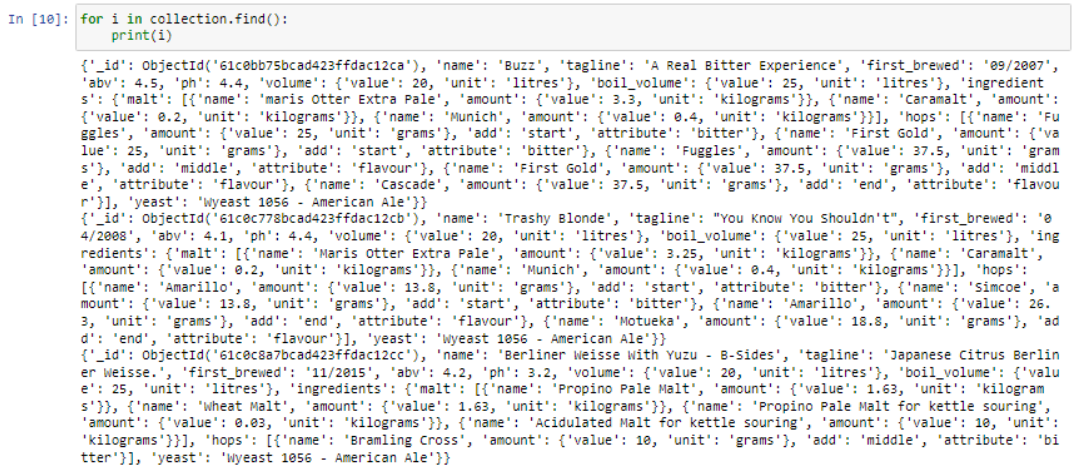
Slika 5: Sučelje Anaconda Navigatora; autorov uradak

U Anaconda Navigatoru se pokreće Notebook, unutar kojeg se otvara nova "bilježnica". Unutar te bilježnice je *importan* pymongo, koji služi za spajanje sa bazom podataka MongoDB.



Slika 6: Spajanje na bazu *appdb* u Jupyter Notebooku; autorov uradak

Spajanje na samu bazu podataka ovdje je izvršeno na način da se varijabli "client" dodaje "MongoClient('localhost', 27017)", gdje je 27017 port na kojem je MongoDB. Varijabli "db" je pridodana baza podataka "appdb", a varijabli "collection" kolekcija "piva" unutar baze podataka "appdb". Kao što se na slici vidi, naredba "db.piva.find()" ne vraća zapise u bazi, već samo kursor.



Slika 7: Dohvaćanje zapisa u bazi *appdb*; autorov uradak

Kako bi se dohvatili zapisi unutar baze, moguće je koristiti iteraciju kao na slici, koja vraća ona piva koja su upisana prethodno u Windows Terminalu.

4.3. MongoDB Database Tools

MongoDB Database Tools je skupina naredbi koje se koriste unutar komandne linije za *exportanje, dumpanje, importanje* i ostale funkcionalnosti. Baza podataka ovog projekta koja sadrži kolekciju "piva" je *dumpana* pomoću ove skupine alata.

Mongodump je naredba korištena za binarni izvoz sadržaja baze "appdb", koji se nalazi u *folderu* "dump", a mongoexport je korišten za izvoz kolekcije "piva" u pivaExport.json.

Baza podataka se može automatski kreirati korištenjem mongoimport naredbe, koja uvozi sadržaj iz već spomenute pivaExport.json datoteke, ili naredbom mongorestore, koja vraća podatke iz datoteka nastalih korištenjem mongodump naredbe.

```
PS C:\Program Files\MongoDB\Tools\100\bin> .\mongodump
2021-12-21T16:57:17.885+0100    writing admin.system.version to dump\admin\system.version.bson
2021-12-21T16:57:17.888+0100    done dumping admin.system.version (1 document)
2021-12-21T16:57:17.889+0100    writing appdb.piva to dump\appdb\piva.bson
2021-12-21T16:57:17.892+0100    done dumping appdb.piva (3 documents)
PS C:\Program Files\MongoDB\Tools\100\bin> .\mongoexport
2021-12-21T17:08:15.584+0100    must specify a collection
2021-12-21T17:08:15.585+0100    try 'mongoexport --help' for more information
PS C:\Program Files\MongoDB\Tools\100\bin> .\mongoexport --collection=piva --db=appdb --out=piva.json
2021-12-21T17:09:03.924+0100    connected to: mongodb://localhost/
2021-12-21T17:09:03.929+0100    exported 3 records
```

Slika 8: Izvoz sadržaja baze podataka pomoću naredbi *mongodump* i *mongoexport*; autorov uradak

5. Primjeri korištenja

Do sada je opisana domena aplikacije ovog projekta, napravljen je teorijski uvod, prikazan je model baze podataka i prikazan je rad u terminalu s MongoDB, u Jupyter Notebooku i rad s MongoDB Database Tools. U ovom poglavlju bit će prikazan osnovni način rada sa MongoDB kolekcijama, odnosno nekoliko upita koje bi inženjer podataka mogao napraviti nad bazom. Upiti će biti prikazani i objašnjeni sa uzlaznom složenosti.

5.1. Jednostavni upiti

Najjednostavniji upiti u MongoDB, osim *.find()*, jesu oni koji vraćaju zapis koji odgovara nekom jednostavnom pravilu. Recimo da želimo, prije svega, vidjeti koliko zapisa uopće postoji u bazi podataka. Upit bi izgledao ovako:

```
In [18]: collection.count_documents({})
Out[18]: 3
```

Slika 9: Upit 1; autorov uradak

Recimo da želimo pronaći pivo s postotkom alkohola od 4.5. Upit i rezultat će izgledati ovako:

```
In [16]: for i in collection.find({"abv": 4.5}):
             print(i)

{'_id': ObjectId('61c0bb75bcad423ffdac12ca'), 'name': 'Buzz', 'tagline': 'A Real Bitter Experience', 'first_brewed': '09/2007',
'abv': 4.5, 'ph': 4.4, 'volume': {'value': 20, 'unit': 'litres'}, 'boil_volume': {'value': 25, 'unit': 'litres'}, 'ingredient
s': {'malt': [{'name': 'maris Otter Extra Pale', 'amount': {'value': 3.3, 'unit': 'kilograms'}}, {'name': 'Caramalt', 'amount':
{'value': 0.2, 'unit': 'kilograms'}}, {'name': 'Munich', 'amount': {'value': 0.4, 'unit': 'kilograms'}}], 'hops': [{'name': 'Fu
ggles', 'amount': {'value': 25, 'unit': 'grams'}, 'add': 'start', 'attribute': 'bitter'}, {'name': 'First Gold', 'amount': {'va
lue': 25, 'unit': 'grams'}, 'add': 'start', 'attribute': 'bitter'}, {'name': 'Fuggles', 'amount': {'value': 37.5, 'unit': 'gram
s'}, 'add': 'middle', 'attribute': 'flavour'}, {'name': 'First Gold', 'amount': {'value': 37.5, 'unit': 'grams'}, 'add': 'middl
e', 'attribute': 'flavour'}, {'name': 'Cascade', 'amount': {'value': 37.5, 'unit': 'grams'}, 'add': 'end', 'attribute': 'flavou
r'}], 'yeast': 'Wyeast 1056 - American Ale'}}
```

Slika 10: Upit 2; autorov uradak

U istom upitu, možemo sakriti nepotrebne atribute na sljedeći način:

```
In [20]: for pivo in collection.find({"abv":4.5}, {"name":1, "abv":1, "ph":1, "tagline":1, "volume":1, "_id":0}):
             print(pivo)

{'name': 'Buzz', 'tagline': 'A Real Bitter Experience', 'abv': 4.5, 'ph': 4.4, 'volume': {'value': 20, 'unit': 'litres'}}
```

Slika 11: Upit 3; autorov uradak

Ako pak želimo pronaći sve pive s postotkom alkohola jednakim ili većim od 4.2, koristimo sljedeći upit:

```
In [24]: for i in collection.find({"abv": {"$gte":4.2}}, {"name":1, "abv":1, "ph":1, "tagline":1, "volume":1, "_id":0}):
             print(i)

{'name': 'Buzz', 'tagline': 'A Real Bitter Experience', 'abv': 4.5, 'ph': 4.4, 'volume': {'value': 20, 'unit': 'litres'}}
{'name': 'Berliner Weisse With Yuzu - B-Sides', 'tagline': 'Japanese Citrus Berliner Weisse.', 'abv': 4.2, 'ph': 3.2, 'volume':
{'value': 20, 'unit': 'litres'}}
```

Slika 12: Upit 4; autorov uradak

5.2. Složeniji upiti

Budući da MongoDB kolekcije sadrže ugniježđene dokumente, bit će prikazan rad s istima. Pretpostavimo da želimo naći vrijednosti atributa "volume" od svakog piva gdje je postotak alkohola jednak ili veći od 4.2. Upit i rezultat izgleda ovako:

```
In [28]: for pivo in collection.find({"abv":{"$gte":4.2}}, {"_id":0, "volume.value":1, "name":1, "abv":1}):
             print(pivo)

         {'name': 'Buzz', 'abv': 4.5, 'volume': {'value': 20}}
         {'name': 'Berliner Weisse With Yuzu - B-Sides', 'abv': 4.2, 'volume': {'value': 20}}
```

Slika 13: Upit 5; autorov uradak

Vidimo da se ugniježđenim atributima pristupa korištenjem sintakse *"<roditelj>.<dijete>"*. Recimo da želimo naći piva kojima ime počinje slovom "B", a vrijednost atributa "boil_volume" je veće od 20.

```
In [31]: for pivo in collection.find({"name":{"$regex":"^B"}, "boil_volume.value":{"$gt":20}},
                                    {"_id":0, "boil_volume.value":1, "name":1}):
             print(pivo)

         {'name': 'Buzz', 'boil_volume': {'value': 25}}
         {'name': 'Berliner Weisse With Yuzu - B-Sides', 'boil_volume': {'value': 25}}
```

Slika 14: Upit 6; autorov uradak

Upit kojim želimo pronaći piva koja nemaju gorkih sastojaka izgleda ovako:

```
In [35]: for pivo in collection.find({"ingredients.hops.attribute":"bitter", "ingredients.hops.attribute":{"$ne":"flavour"}},
                                    {"name":1, "ingredients.hops.name":1, "_id":0}):
             print(pivo)

         {'name': 'Berliner Weisse With Yuzu - B-Sides', 'ingredients': {'hops': [{'name': 'Bramling Cross'}]}}
```

Slika 15: Upit 7; autorov uradak

5.3. MapReduce

MapReduce je paradigma procesiranja podataka koja se koristi za sažimanje velikih količina podataka u korisne agregirane rezultate [8]. Implementira se pomoću funkcija u JavaScriptu. Primjer upita koji mapira i reducira podatke u MongoDB može biti sljedeći:

```
In [61]: from bson.code import Code
         map = Code("function() {"
                    " emit( this.ph, this.abv );"
                    "}")
         reduce = Code("function(key, values) {"
                       "return Array.sum(values)"
                       "}")
         upit = collection.map_reduce(map, reduce, "rezultat")
         for i in upit.find():
             print(i)

---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-61-44f6a67ca692> in <module>
      6                "return Array.sum(values)"
      7                "}")
----> 8 upit = collection.map_reduce(map, reduce, "rezultat")
      9 for i in upit.find():
     10     print(i)

~\anaconda3\lib\site-packages\pymongo\collection.py in __call__(self, *args, **kwargs)
   2582                                 "exists." %
   2583                                 self.__name)
-> 2584         raise TypeError("'Collection' object is not callable. If you meant to "
   2585                         "call the '%s' method on a 'Collection' object it is "
   2586                         "failing because no such method exists." %

TypeError: 'Collection' object is not callable. If you meant to call the 'map_reduce' method on a 'Collection' object it is fai
ling because no such method exists.
```

Slika 16: Upit 8; autorov uradak

Međutim, kako se ovdje može vidjeti, mapReduce je zastarjela (*deprecated*) operacija, te ju pyMongo više ne podržava. Za ovaj upit trebamo se vratiti u terminal.

```
appdb> var map = function () { emit(this.ph, this.abv);};

appdb> var reduce = function (key, value) { return Array.sum(value); };

appdb> db.piva.mapReduce(map, reduce, {out: "map_reduce"})
{ result: 'map_reduce', ok: 1 }
appdb> db.map_reduce.find().sort({_id: 1})
[ { _id: 3.2, value: 4.2 }, { _id: 4.4, value: 8.6 } ]
```

Slika 17: MapReduce u terminalu; autorov uradak

U ovom upitu se postavlja da varijabla "map" koja služi kao funkcija mapiranja u *mapReduce* te varijabla "reduce" koja služi kao funkcija reduciranja. Funckija mapReduce stvara novu kolekciju koja je ovdje nazvana "map_reduce" koju se može pretraživati koristeći *.find()*. Upit agregira vrijednosti postotka alkohola u pivu u odnosu na njihovu pH vrijednost. Budući da postoji samo jedno pivo sa pH vrijednosti od 3.2 i abv od 4.2, vraća se dokument sa "identifikatorom" 3.2 i vrijednošću 4.2. Ostala dva piva imaju pH vrijednosti od 4.4, a abv 4.5 i 4.1, što vraća dokument sa "identifikatorom" 4.4 i vrijednošću 4.5+4.1 = 8.6.

6. Zaključak

U ovom je projektnom radu opisana važnost analize velikih količina podataka, postupak i definicija iste te aplikacijska domena ovog rada. Ukratko je objašnjen polustrukturirani model baza podataka te njegove kolekcije i dokumenti. Navedene su neke od prednosti i nedostataka polustrukturiranog modela nad drugim modela baza podataka te primjeri tipova podataka u polustrukturiranom modelu. Prikazan je model baze podataka, implementacija aplikacije i baze podataka tj. rad s MongoDB u terminalu te vršenje upita u Jupyter Notebooku, a smišljeni su i prikazani upiti uzlazne složenosti te procedura *mapReduce*.

MongoDB, Jupyter Notebook i Python su često korišteni alati pri radu s podacima, a rastući broj korisnika interneta znači i rastući broj podataka koje treba spremati, analizirati i koristiti. U svrhu lakog i prostorno efikasnog spremanja podataka (JSON datoteke), kao i apstraktan rad s bazama podataka koristeći Python i njegove biblioteke poput Pandas zasigurno čini ove alate izuzetno privlačnima svakome tko želi postati inženjer podataka. Autorovo osobno mišljenje je da je MongoDB, usprkos svojih nedostataka, bolja verzija relacijskih baza podataka te je manjak fokusa na rad s MongoDB i pratećim alatima jedna od većih grešaka današnjih fakulteta u grani IKT-a.

7. Literatura

[1]„Big Data Analytics: What It Is, How It Works, Benefits, And Challenges“, Tableau. https://www.tableau.com/learn/articles/big-data-analytics (pristupljeno pros. 16, 2021).

[2]D. Suciu, „Semi-Structured Data Model“, Encyclopedia of Database Systems, str. 2601–2605, 2009, https://link.springer.com/referenceworkentry/10.1007/978-0-387-39940-9_337 (pristupljeno pros.16, 2021).

[3]„What is Semi-structured data?“, GeeksforGeeks, tra. 12, 2019. https://www.geeksforgeeks.org/wha is-semi-structured-data/ (pristupljeno pros. 16, 2021).

[4]„What Is Semi-Structured Data?“, MonkeyLearn Blog, stu. 16, 2020. https://monkeylearn.com/blog/s structured-data/ (pristupljeno pros. 16, 2021).

[5]„JSON Example“. http://www.json.org/example.html (pristupljeno pros. 17, 2021).

[6]„MongoDB - Data Modelling“. https://www.tutorialspoint.com/mongodb/mongodb_data_modeling.htr (pristupljeno pros. 17, 2021).

[7]„Punk API: Brewdog’s DIY Dog as an API“. https://punkapi.com (pristupljeno pros. 20, 2021).

[8]„Map-Reduce — MongoDB Manual“, https://github.com/mongodb/docs-bi-connector/blob/DOCSP-3279/source/index.txt. https://docs.mongodb.com/manual/core/map-reduce/ (pristupljeno pros. 22, 2021).

Poveznica na Overleaf projekt: https://www.overleaf.com/project/61bb593898df0612e4ba76c6