

Міністерство освіти і науки України  
Національний університет «Львівська політехніка»  
Інститут комп'ютерних наук та інформаційних технологій  
Кафедра «Системи штучного інтелекту»

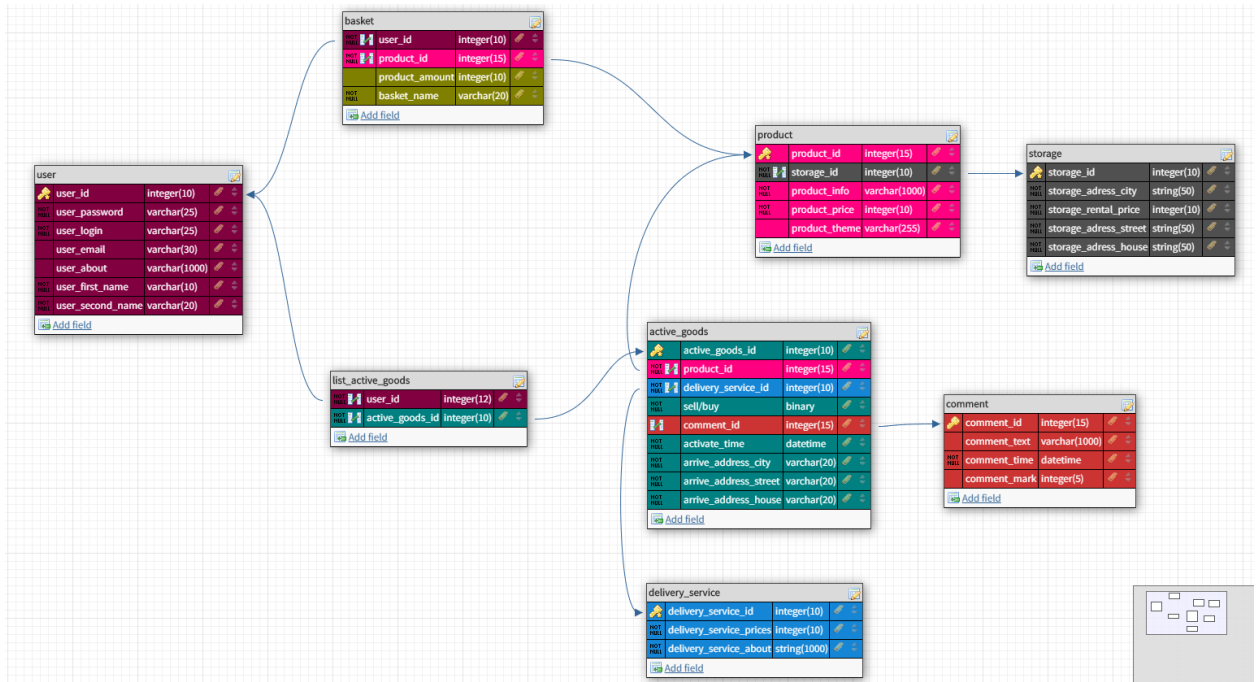


**Лабораторна робота №14**  
**з дисципліни: «ОБДЗ»**

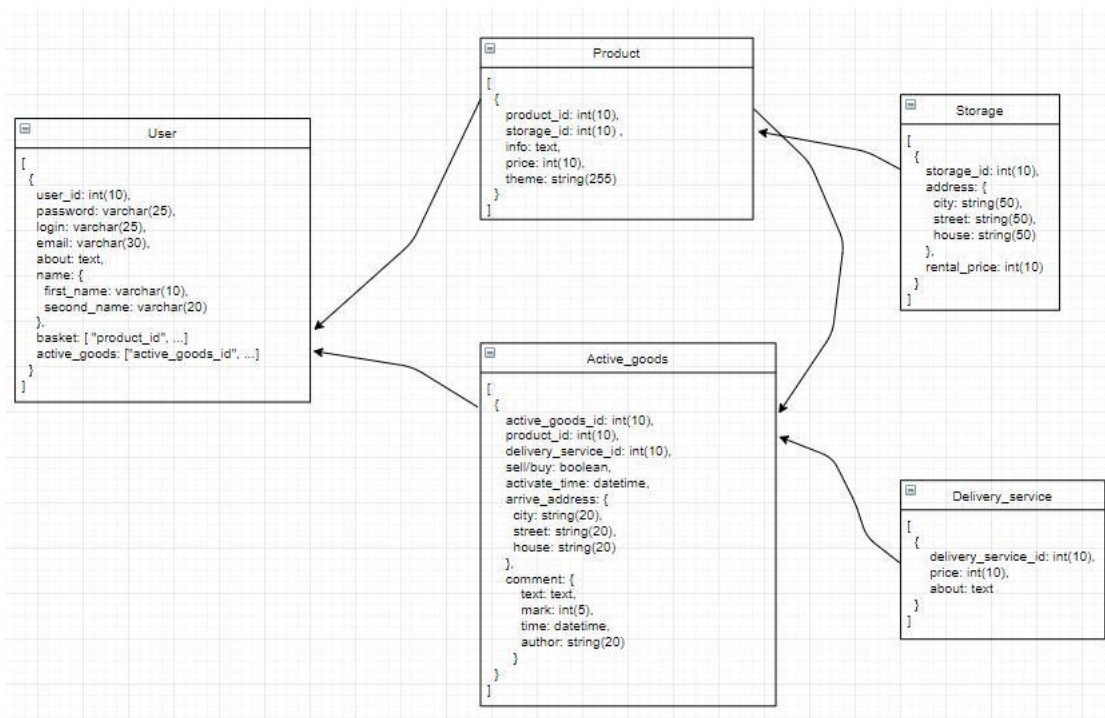
*Виконав студент*  
*групи КН-208*  
*Жеребецький Олег*  
*Прийняла:*  
*асистент*  
*Якимішин Х.М.*

*Львів-2020*

## Схема з лаб 1.



## Хід роботи.



### 1. Для початку скачаємо драйвер для Python для роботи з mongodb

```
(venv) C:\01eh\Programs\Coding\db>python -m pip install pymongo
```

```
Collecting pymongo
```

```
Downloading https://files.pythonhosted.org/packages/dc/9b/6791f7219f3573bfaa2251da4d814f4fbc49f0bbb258df1e08f7d89a7b85/pymongo-3.10.1.tar.gz (715kB)
100% |████████████████████████████████████████| 716kB 107kB/s
```

```
Installing collected packages: pymongo
```

```
Running setup.py install for pymongo ... done
```

```
Successfully installed pymongo-3.10.1
```

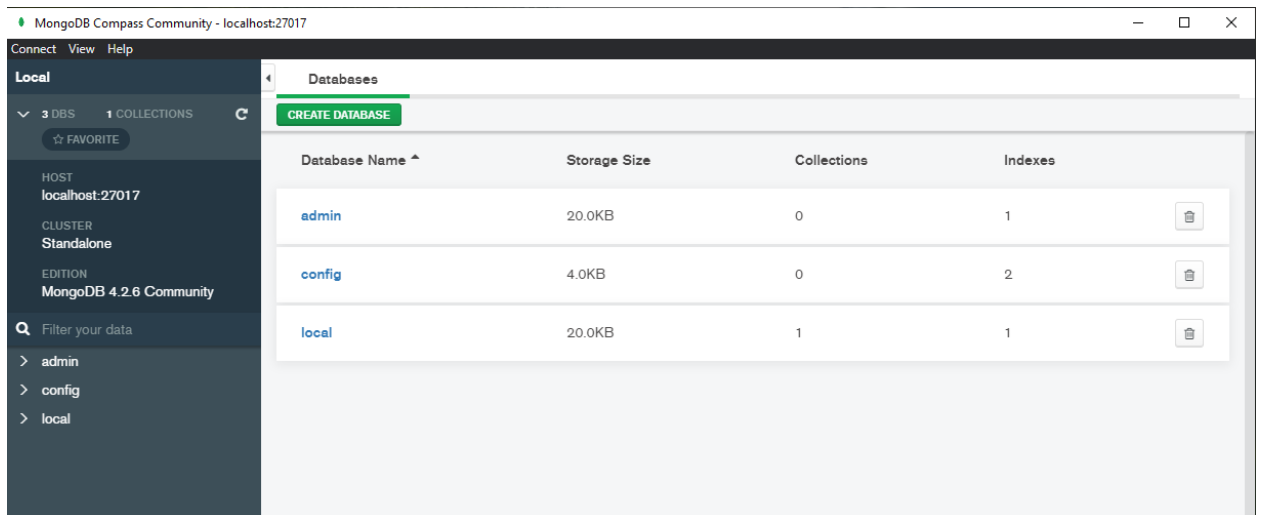
2. Якщо ми захочемо подивитись чи створилась база даних то ми цього не побачимо, оскільки вона створиться при заповненні першими даними.

**Важливо:** У MongoDB база даних не створюється, поки вона не отримає вміст!

```
import pymongo

myclient = pymongo.MongoClient("mongodb://localhost:27017/")

mydb = myclient["databaseLab"]
```



```
dblist = myclient.list_database_names()
if "databaseLab" in dblist:
    print("The database exists.")
else:
    print("The database not exists.")
```

The database not exists.

3. Створимо колекцію. Насправді вона реально створиться при заповненні першими даними

**Важливо:** у MongoDB колекція не створюється, поки вона не отримає вміст!

```
myclient = pymongo.MongoClient("mongodb://localhost:27017/")
mydb = myclient["mydatabase"]
```

```
mycol = mydb["User"]
```

```
collist = mydb.list_collection_names()
if "User" in collist:
    print("The collection exists.")
else:
    print("The collection not exists.")
```

```
C:\Oleh\Programs\Coding\db\venv\Scripts\python.exe C:/Oleh/Programs/Coding/db/db_12.py
The collection not exists.
```

```
Process finished with exit code 0
```

#### 4. Заповнимо першу табличку одним рядком даних

```
myclient = pymongo.MongoClient("mongodb://localhost:27017/")
mydb = myclient["mydatabase"]
User = mydb["User"]

value = { "password": "12345678", "login": "oleh_galis", "email": "olehvel@gmail.com", "about": "About user1",
          "name": { "first_name": "Oleh", "second_name": "Zherebetskiy"}, "basket": [1], "active_goods": [1] }

x = User.insert_one(value)


print(x.inserted_id)
```

#### 5. Виведемо \_id яке mongoDB само згенерувало

```
C:\Oleh\Programs\Coding\db\venv\Scripts\python.exe C:/Oleh/Programs/Coding/db/db_13.py
5ebcd2ab4f48db211c89723d
```

```
Process finished with exit code 0
```

#### 6. Оскільки ми заповнили перші дані то переглянемо бд і колекцію в програмі Compas

mydatabase	20.0KB	1	1	

Collections

CREATE COLLECTION

Collection Name ^	Documents	Avg. Document Size	Total Document Size	Num. Indexes	Total Index Size	Properties
User	1	227.0 B	227.0 B	1	20.0 KB	

mydatabase.User Documents

mydatabase.User

DOCUMENTS 1 TOTAL SIZE 227B AVG. SIZE 227B INDEXES 1 TOTAL SIZE 20.0KB AVG. SIZE 20.0KB

Documents Aggregations Explain Plan Indexes

FILTER OPTIONS FIND RESET ...

ADD DATA VIEW {}

Displaying documents 1 - 1 of 1 REFRESH

```
> { "_id": ObjectId("Sebcd2ab4f48db211c89723d"),
  "password": "12345678",
  "login": "oleh_galis",
  "email": "olehvel@gmail.com",
  "about": "About user1",
  "name": { "first_name": "Oleh", "second_name": "Zherebetskiy" },
  "basket": [1],
  "active_goods": [1] }
```

- Дані зберігаються у вигляді джейсона, який зберігається в бінарному представленні

7. Введемо багато рядків у таблицю.

```
listValue = [
    { "_id": 1, "password": "12345678", "login": "oleh_galis", "email": "olehvel@gmail.com", "about": "About user1",
      "name": { "first_name": "Oleh", "second_name": "Zherebetskiy" }, "basket": [1], "active_goods": [1] },
    { "_id": 2, "password": "12345678", "login": "msdfdsf", "email": "sgdfg@gmail.com", "about": "About user12",
      "name": { "first_name": "Maria", "second_name": "Rizhko" }, "basket": [2,1], "active_goods": [2,6] },
    { "_id": 3, "password": "s123c", "login": "rtbdgbhgvvf", "email": "oldfgehvel@gmail.com", "about": "About user1fg",
      "name": { "first_name": "avre", "second_name": "Zherebearvgavtskiy" }, "basket": [3,1], "active_goods": [3] },
    { "_id": 4, "password": "123ewctw45cer678", "login": "dvdfhghsh", "email": "ergagrar@gmail.com", "about": "About user1fg",
      "name": { "first_name": "Olesdfh", "second_name": "Zhersdfsfdebetskiy" }, "basket": [4,2], "active_goods": [4] },
    { "_id": 5, "password": "rccewrcew", "login": "dghgfhf", "email": "olehvdfel@gmail.com", "about": "About user1dsf",
      "name": { "first_name": "Olesdrh", "second_name": "Zhereervdfvbetkiy" }, "basket": [5,3,4], "active_goods": [5] },
]

x = User.insert_many(listValue)

print(x.inserted_ids)
```

8. Виведемо масив ід які ми задали при внесенні даних

[1, 2, 3, 4, 5]

9. Спробуємо дістати дані з певної колекції. Для початку тільки один документ(перший)

У MongoDB ми використовуємо методи **find** and **findOne** для пошуку даних у колекції.

Так само, як **оператор SELECT** використовується для пошуку даних у таблиці в базі даних MySQL.

```
import pymongo
```

```
myclient = pymongo.MongoClient("mongodb://localhost:27017/")
mydb = myclient["mydatabase"]
User = mydb["User"]
```

```
x = User.find_one()
```

```
print(x)
```

```
C:\Oleh\Programs\Coding\db\venv\Scripts\python.exe C:\Oleh\Programs\Coding\db\db_12.py
```

```
{'_id': ObjectId('5ebcd2ab4f48db211c89723d'), 'password': '12345678', 'login': 'oleh_galis', 'email': 'olehvel@gmail.com', 'about': 'About user1', 'name': {'first_name': 'Oleh'}}
```

## 10. Тепер виведемо усі документи з колекції

```
for x in User.find():
    print(x)
```

```
C:\Oleh\Programs\Coding\db\venv\Scripts\python.exe C:\Oleh\Programs\Coding\db\db_12.py
```

```
{'_id': ObjectId('5ebcd2ab4f48db211c89723d'), 'password': '12345678', 'login': 'oleh_galis', 'email': 'olehvel@gmail.com', 'about': 'About user1', 'name': {'first_name': 'Oleh', 'second_name': 'Zherebetskiy'}, 'basket': []},
{'_id': 1, 'password': '12345678', 'login': 'oleh_galis', 'email': 'olehvel@gmail.com', 'about': 'About user1', 'name': {'first_name': 'Oleh', 'second_name': 'Zherebetskiy'}, 'basket': []},
{'_id': 2, 'password': '12345678', 'login': 'msdfdsf', 'email': 'sgdfg@gmail.com', 'about': 'About user12', 'name': {'first_name': 'Maria', 'second_name': 'Rizhko'}, 'basket': [2, 1]},
{'_id': 3, 'password': 's123c', 'login': 'rtbdgbhgvvf', 'email': 'oldfgehvel@gmail.com', 'about': 'About user1fg', 'name': {'first_name': 'avre', 'second_name': 'Zherebearvgavtskiy'}, 'basket': [2, 1]},
{'_id': 4, 'password': '123ewctw45cer678', 'login': 'dvdfhghsh', 'email': 'ergagrar@gmail.com', 'about': 'About user1fg', 'name': {'first_name': 'Olesdfh', 'second_name': 'Zhersdfsfdebetskiy'}, 'basket': [2, 1]},
{'_id': 5, 'password': 'rccewrcew', 'login': 'dghgfhd', 'email': 'olehvsdfel@gmail.com', 'about': 'About user1dsf', 'name': {'first_name': 'Olesdrh', 'second_name': 'Zhereervdfvbetkiy'}, 'basket': [2, 1]}
```

## 11. Задамо в пошуці поля які ми хочемо бачити(1)

```
for x in User.find({}, {"_id": 0, "name": 1, "login": 1}):
    print(x)
```

```
{'login': 'oleh_galis', 'name': {'first_name': 'Oleh', 'second_name': 'Zherebetskiy'}}
{'login': 'oleh_galis', 'name': {'first_name': 'Oleh', 'second_name': 'Zherebetskiy'}}
{'login': 'msdfdsf', 'name': {'first_name': 'Maria', 'second_name': 'Rizhko'}}
{'login': 'rtbdgbhgvvf', 'name': {'first_name': 'avre', 'second_name': 'Zherebearvgavtskiy'}}
{'login': 'dvdfhghsh', 'name': {'first_name': 'Olesdfh', 'second_name': 'Zhersdfsfdebetskiy'}}
{'login': 'dghgfhd', 'name': {'first_name': 'Olesdrh', 'second_name': 'Zhereervdfvbetkiy'}}
```

## 12. Виведемо все крім \_id

```
for x in User.find({}, {"_id": 0}):
    print(x)
```

```
{'password': '12345678', 'login': 'oleh_galis', 'email': 'olehvel@gmail.com', 'about': 'About user1', 'name': {'first_name': 'Oleh', 'second_name': 'Zherebetskiy'}, 'basket': []},
{'password': '12345678', 'login': 'oleh_galis', 'email': 'olehvel@gmail.com', 'about': 'About user1', 'name': {'first_name': 'Oleh', 'second_name': 'Zherebetskiy'}, 'basket': []},
{'password': '12345678', 'login': 'msdfdsf', 'email': 'sgdfg@gmail.com', 'about': 'About user12', 'name': {'first_name': 'Maria', 'second_name': 'Rizhko'}, 'basket': [2, 1]},
{'password': 's123c', 'login': 'rtbdgbhgvvf', 'email': 'oldfgehvel@gmail.com', 'about': 'About user1fg', 'name': {'first_name': 'avre', 'second_name': 'Zherebearvgavtskiy'}, 'basket': [2, 1]},
{'password': '123ewctw45cer678', 'login': 'dvdfhghsh', 'email': 'ergagrar@gmail.com', 'about': 'About user1fg', 'name': {'first_name': 'Olesdfh', 'second_name': 'Zhersdfsfdebetskiy'}, 'basket': [2, 1]},
{'password': 'rccewrcew', 'login': 'dghgfhd', 'email': 'olehvsdfel@gmail.com', 'about': 'About user1dsf', 'name': {'first_name': 'Olesdrh', 'second_name': 'Zhereervdfvbetkiy'}, 'basket': [2, 1]}
```

## 13. Проведемо при пошуці відбір тільки тих документів в яких логін саме такий...

```
mydb = myclient["mydatabase"]
User = mydb["User"]
```

```
myquery = { "login": "oleh_galis" }
mydoc = User.find(myquery)
```

```
for x in mydoc:
    print(x)
```

```
{ '_id': ObjectId('5ebcd2ab4f48db211c89723d'), 'password': '12345678', 'login': 'oleh_galis', 'email': 'olehvel@gmail.com', 'about': 'About user1', 'name': {'first_name': 'Oleh', 'second_name': 'Zherebetskiy'}, 'password': '12345678', 'login': 'oleh_galis', 'email': 'olehvel@gmail.com', 'about': 'About user1', 'name': {'first_name': 'Oleh', 'second_name': 'Zherebetskiy'}, ... }
```

## 14. Використаємо регулярний вираз для відбору даних

```
myquery = { "login": { "$regex": "^o" } }
```

```
mydoc = User.find(myquery)
```

```
for x in mydoc:
    print(x)
```

```
{ '_id': ObjectId('5ebcd2ab4f48db211c89723d'), 'password': '12345678', 'login': 'oleh_galis', 'email': 'olehvel@gmail.com', 'about': 'About user1', 'name': {'first_name': 'Oleh', 'second_name': 'Zherebetskiy'}, 'password': '12345678', 'login': 'oleh_galis', 'email': 'olehvel@gmail.com', 'about': 'About user1', 'name': {'first_name': 'Oleh', 'second_name': 'Zherebetskiy'}, ... }
```

## 15. Виведемо тільки одне поле посортувавши по ньому по зростанню

```
mydb = myclient["mydatabase"]
User = mydb["User"]
```

```
mydoc = User.find({}, {"_id": 0, "login": 1}).sort("login", -1)
```

```
for x in mydoc:
    print(x)
```

```
C:\Oleh\Programs\Coding\db\
{'login': 'rtbdgbhgvfgf'}
{'login': 'oleh_galis'}
{'login': 'oleh_galis'}
{'login': 'msdfdsf'}
{'login': 'dvdfhghsh'}
{'login': 'dghgfhhd'}
```

## 16. Видалимо всі дані з таблиці

```
mydb = myclient["mydatabase"]
User = mydb["User"]
```

```
x = User.delete_many({})
```

```
print(x.deleted_count, " documents deleted.")
```

## 17.Видалимо тільки ті дані що відповідають певній умові

```
mydb = myclient["mydatabase"]
User = mydb["User"]
```

```
myquery = { "login": { "$regex": "^o" } }
```

```
x = User.delete_many(myquery)
```

```
print(x.deleted_count, " documents deleted.")
```

## 18.Виведемо скільки документів було видалено

```
C:\Oleh\Programs\Coding\db\venv\Scr
2 documents deleted.
```

```
Process finished with exit code 0
```

## 19.Видалимо колекцію

```
mydb = myclient["mydatabase"]
mycol = mydb["customers"]
```

```
mycol.drop()
```

## 20.Оновимо Значення в одному(першому) документі певної колекції

**Примітка:** Якщо в запиті знайдено більше одного запису, оновлюється лише перше виникнення.

```
update_one()
```



```

    'id': 2, 'password': '12345678', 'login': 'msbtrst', 'email': 'sgutge@gmail.com', 'about': 'About user12', 'name': {'first_name': 'Nikita', 'second_name': 'Kizhko', 'last_name': 'Dobek'},
    'id': 3, 'password': 's123c', 'login': 'rtbdghgvgyf', 'email': 'oldfghvel@gmail.com', 'about': 'About user1fg', 'name': {'first_name': 'avre', 'second_name': 'Zherebarnygav', 'last_name': 'Dobek'},
    'id': 4, 'password': '9ccwctwa45cer678', 'login': 'dvdfghsh', 'email': 'ergagrar@gmail.com', 'about': 'About user1fg', 'name': {'first_name': 'Olesdfh', 'second_name': 'Zherebarnygav', 'last_name': 'Dobek'},
    'id': 5, 'password': 'rccvercw', 'login': 'dghghfd', 'email': 'olehvsdfel@gmail.com', 'about': 'About user1dsf', 'name': {'first_name': 'Olesdrh', 'second_name': 'Zhereverid', 'last_name': 'Dobek'},
    'id': 1, 'password': '12345678', 'login': 'OLEH GALIS', 'email': 'olehvel@gmail.com', 'about': 'About user1', 'name': {'first_name': 'Oleh', 'second_name': 'Zherebetskiy', 'last_name': 'Dobek'}
  ],
  'password': '12345678', 'login': 'OLEH GALIS', 'email': 'olehvel@gmail.com', 'about': 'About user1', 'name': {'first_name': 'Oleh', 'second_name': 'Zherebetskiy', 'last_name': 'Dobek'}
}

```

```
for x in User.find():
    print(x)
```

```
{'_id': 2, 'password': 'OLEH_GALIS', 'login': 'msdfdsf', 'email': 'sgdfg@gmail.com', 'about': 'OLEH_GALIS', 'name': {'first_name': 'Maria', 'second_name': 'Rizhko'}}, 'basket'
{'_id': 3, 'password': 'OLEH_GALIS', 'login': 'rtbdgbhgvgf', 'email': 'oldfgehvel@gmail.com', 'about': 'OLEH_GALIS', 'name': {'first_name': 'avre', 'second_name': 'Zherebear'
{'_id': 4, 'password': 'OLEH_GALIS', 'login': 'dvdfhghsh', 'email': 'ergagrar@gmail.com', 'about': 'OLEH_GALIS', 'name': {'first_name': 'Olesdfh', 'second_name': 'Zhersdfsd'
{'_id': 5, 'password': 'rcccewrcew', 'login': 'dghghfd', 'email': 'olehvsdfel@gmail.com', 'about': 'About user1dsf', 'name': {'first_name': 'Olesdrh', 'second_name': 'Zhereer'
{'_id': 1, 'password': 'OLEH_GALIS', 'login': 'OLEH_GALIS', 'email': 'olehvel@gmail.com', 'about': 'OLEH_GALIS', 'name': {'first_name': 'Oleh', 'second_name': 'Zherebetskiy'}
```

22. Якщо ми хочемо вивести певне поле об'єкта який є полем то зробимо ось так -

```
mydb = myclient["mydatabase"]
User = mydb["User"]

for x in User.find({}, {"_id": 0, "name": 1}):
    print({"name": {"first_name": x['_name']['first_name']}})

C:\Oleh\Programs\Coding\db\venv\Script
{'name': {'first_name': 'Maria'}}
{'name': {'first_name': 'avre'}}
{'name': {'first_name': 'Olesdfh'}}
{'name': {'first_name': 'Olesdrh'}}
{'name': {'first_name': 'Oleh'}}
```

23. Обмежимо пошук лише до двох значень

```
mydb = myclient["mydatabase"]
User = mydb["User"]

for x in User.find({}, {"_id": 0, "name": 1}).limit(2):
    print({"name": {"first_name": x['_name']['first_name']}})

C:\Oleh\Programs\Coding\db\venv\Scripts\py
{'name': {'first_name': 'Maria'}}
{'name': {'first_name': 'avre'}}
```

24. В результаті заглянемо в програму Comras де можемо легко побачити усі документи певних колекцій

mydatabase.User

DOCUMENTS 5

TOTAL SIZE 1.1KB

AVG. SIZE 233B

INDEXES 1

TOTAL SIZE 36.0KB

AVG. SIZE 36.0KB

Documents

Aggregations

Explain Plan

Indexes

FILTER

OPTIONS

FIND

RESET

...

ADD DATA

VIEW

Displaying documents 1 - 5 of 5

REFRESH

\_id: 1

password: "OLEH\_GALIS"

login: "OLEH\_GALIS"

email: "olehvel@gmail.com"

about: "OLEH\_GALIS"

> name: Object

> basket: Array

> active\_goods: Array

\_id: 2

password: "OLEH\_GALIS"

login: "msdfdsf"

email: "sgdfg@gmail.com"

about: "OLEH\_GALIS"

> name: Object

> basket: Array

> active\_goods: Array

\_id: 3

password: "OLEH\_GALIS"

login: "rtbdghgvvf"

email: "oldfgehvel@gmail.com"

about: "OLEH\_GALIS"

> name: Object

> basket: Array

> active\_goods: Array

\_id: 4

password: "OLEH\_GALIS"

login: "dvdfhghsh"

email: "ergagran@gmail.com"

about: "OLEH\_GALIS"

> name: Object

> basket: Array

> active\_goods: Array

\_id: 5

password: "rccewrcew"

login: "dghghfd"

11:13

14.05.2020

CREATE COLLECTION

25. Використаємо умовні операції для задання умов пошуку

```
mydb = myclient["mydatabase"]
User = mydb["User"]

for x in User.find({"_id": {"$lt": 3}}):
    print(x)
```

```
{ '_id': 1, 'password': 'OLEH_GALIS', 'login': 'OLEH_GALIS', 'email': 'olehvel@gmail.com', 'about': 'OLEH_GALIS', 'name': {'first_name': 'Oleh', 'second_name': 'Zherebetskiy'},
  '_id': 2, 'password': 'OLEH_GALIS', 'login': 'msdfdsf', 'email': 'sgdfg@gmail.com', 'about': 'OLEH_GALIS', 'name': {'first_name': 'Maria', 'second_name': 'Rizhko'}, 'basket':
```

26. Створимо новий індекс для поля login і задамо йому значення унікальності

```
mydb = myclient["mydatabase"]
User = mydb["User"]

#User.drop_index("login 1")
User.create_index("login", unique=True)
for x in User.index_information():
    print(x)
```

```
{ '_id_': 1, 'login_1': 1 }
'_id_'
'login_1'
```

## Відповіді на питання

### 1. Назвати основні типи баз даних

- “ключ-значення”
- Документорієнтовані
- Графи
- Об’єктно-орієнтовані бази даних

### 2. Назвати переваги та недоліки використання баз даних NoSQL

- Мова програмування в NoSQL не уніфікована, кожна бд може мати свій синтаксис який різко відрізняється від інших, це ускладнює роботу з ними.
- SQL – мова запитів для роботи з даними, бд має чітку структуру, якщо робити зміни в структурі – можна легко поламати усю бд
- NoSQL – можна створювати документи не задаючи їм структуру взагалі, кожен документ може мати свою структуру, може бути свій синтаксис в кожній бд, можемо добавляти поля прямо під час роботи з даними

- Масштабування
  - SQL – вертикально масштабована
  - NoSQL – горизонтально масштабована
  - Це дає можливість розмістити NoSQL на декількох серверах, що дасть якісну масштабованість
- Структура
  - SQL – таблиці, це добре для реляційних бд
  - NoSQL – документи, пари ключ-значення, графи
- В NoSQL – структура міняється дуже легко
- В NoSQL- гірша зв'язність даних
- NoSQL – можуть програмувати люди які не знають SQL і не мають глибоких знань про бд
- NoSQL - Швидкість простих операцій
- SQL – перевірена часом
- SQL – доступна на більшості відомих платформах

### 3. Надати характеристику СУБД MongoDB

Документо-орієнтована система керування базами даних з відкритим кодом. Не потребує чіткої структури документів. Середнє між швидкістю і масштабованістю. Підтримує зберігання даних в JSON –подібному форматі, має досить гнучку мову для формування запитів. Може створювати індекси для збережених атрибутів. Ефективно зберігає великі бінарні об'єкти.

### 4. Операції вставки даних

Приклади наведені вище в пунктах 4-8.

- `db.persons.insert({"name": "Tom", "age": "28", "languages": ["english", "spanish"]})`

- `db` – наша база даних
- `persons` – наша таблиця
- `insert` – команда додавання документу в колекцію
- `({"name": "Tom", "age": "28", "languages": ["english", "spanish"]})` –дані які ми додаємо як новий документ колекції

- `db.persons.update({ name : "Eugene", age: "29"}, {"age": {"$set:"30"} })`

- `update` – команда оновлення
- `{ name : "Eugene", age: "29" }` - умова за якою ми шукаємо який документ потрібно оновити

- \$set – модифікатор, позначає що ми хочемо встановити значення якогось ключа
- (якщо нема такого поля воно створиться)
- {"age": {\$set:"30"}} дані які ми хочемо встановити
  - db.persons.update({ name : "Tom"}, {\$push: {languages: "ukrainian "}})
- \$push – додавання елементу в масив
  - db.persons.update({ name : "Tom"}, {\$addToSet: {languages: "ukrainian "}})
- \$addToSet - працює по аналогії \$push тільки з перевіркою на унікальність
  - db.persons.update({ name : "Tom"}, {\$addToSet: {languages: {\$each: ["ukrainian", "spanish", "italian"]}}})

## 5. Операції оновлення даних

Приклади наведені вище в пунктах 20-21.

- db.persons.save({"name": "Eugene", "age" : "29", languages: ["english", "german", "spanish"]})
- save – зберігає зі змінами
  - db.persons.update({ name : "Tom"}, {"name": "Tom", "age" : "25", "married" : false}, {upsert: true})
  - db.persons.update({ name : "Tom"}, {"name": "Tom", "age" : "25", "married" : false}, {upsert: true})
  - db.persons.update({ name : "Tom"}, {\$push: {languages: "ukrainian "}})
  - db.persons.update({ name : "Tom"}, {\$inc: {salary: 100}})
- \$inc – для збільшення , наприклад щоб збільшити вік користувача

## 6. Операції знищення даних

Приклади наведені вище в пунктах 16-19.

- db.persons.update({ name : "Tom"}, {\$unset: {salary: 1}})
- \$unset – якраз знищення даних
  - db.persons.update({ name : "Tom"}, {\$unset: {salary: 1, age: ""}})

- `db.persons.remove({name : "Tom"})`
- `db.persons.remove({name : /T\w+/i})`
- `db.persons.remove({age: {$lt : 30}})`
- `db.persons.remove({name : "Tom"}, true)`
- `db.persons.drop()`
- `db.dropDatabase()`
- `db.persons.update({name : "Tom"}, {$pop: {languages: 1}})`

- `$pop` видалення елемента з кінця

## 7. Умовні оператори

Приклади наведені вище в пунктах 25.

- `$eq` - співпадає
  - `$ne` – не співпадає
  - `$gt` – більше ніж
  - `$lt` – менше ніж
  - `$gte` – більше рівне
  - `$lte` – менше рівне
  - `$in` – масив значень, одне з яких мусить містити поле документа
  - `$nin` – масив значень які мусить не бути в полі документа
- 
- `db.persons.find ({age: {$lt : 30}})`
  - `db.persons.find ({age: {$gt : 30}})`

## 8. Операції керування індексами

Приклади наведені вище в пунктах 26.

- `db.persons.ensureIndex({"name" : 1})`
- `db.persons.createIndex({"name" : 1}, {"unique" : true})`

- `createIndex()` – ф-ція створення створення індексу

- `db.persons.ensureIndex({"name" : 1, "age" : 1}, {"unique" : true})`
- `db.mycol.getIndexes()`
- `db.system.indexes.find()`

- знайти усі індекси

- `db.persons.dropIndex("name_1")`

- `dropIndex()` – ф-ція видалення індексу

## 9. Пошук даних

Приклади наведені вище в пунктах 9-15.

- З умовними операторами
  - `db.persons.find ({age: {$lt : 30}})`
- `{age: {$lt : 30}}` - умова за якою проводиться пошук
  - `db.persons.find ({age: {$gt : 30}})`
- З певними операторами
  - `db.persons.find ({age: {$in : [22, 32]}})`
  - `db.persons.find ({age: {$nin : [22, 32]}})`
  - `db.persons.find ({age: {$all : [22, 32]}})`
  - `db.persons.find ({age: {$all : [22]}})`
  - `db.persons.find ({languages: {$all : ["english", "french"]}})`
  - `db.persons.find ({$or : [{name: "Tom"}, {age: "22"}]})`
  - `db.persons.find ({name: "Tom", $or : [{age: "22"}, {languages: "german"}]})`
  - `db.persons.find ({languages: {$size:2}})`
  - `db.persons.find ({company: {$exists:true}})`

## 10. Можливості документних БД

- Узгодження даних
- Транзакції
- Доступність
- Можливості запитів
- масштабування

### Висновок

На цій лабораторній я ознайомився із одною із найпопулярніших документо-орієнтованою базою даних NoSQL – MongoDB. Виконав кілька найпопулярніших основних операцій над документами та їх колекціями.