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*

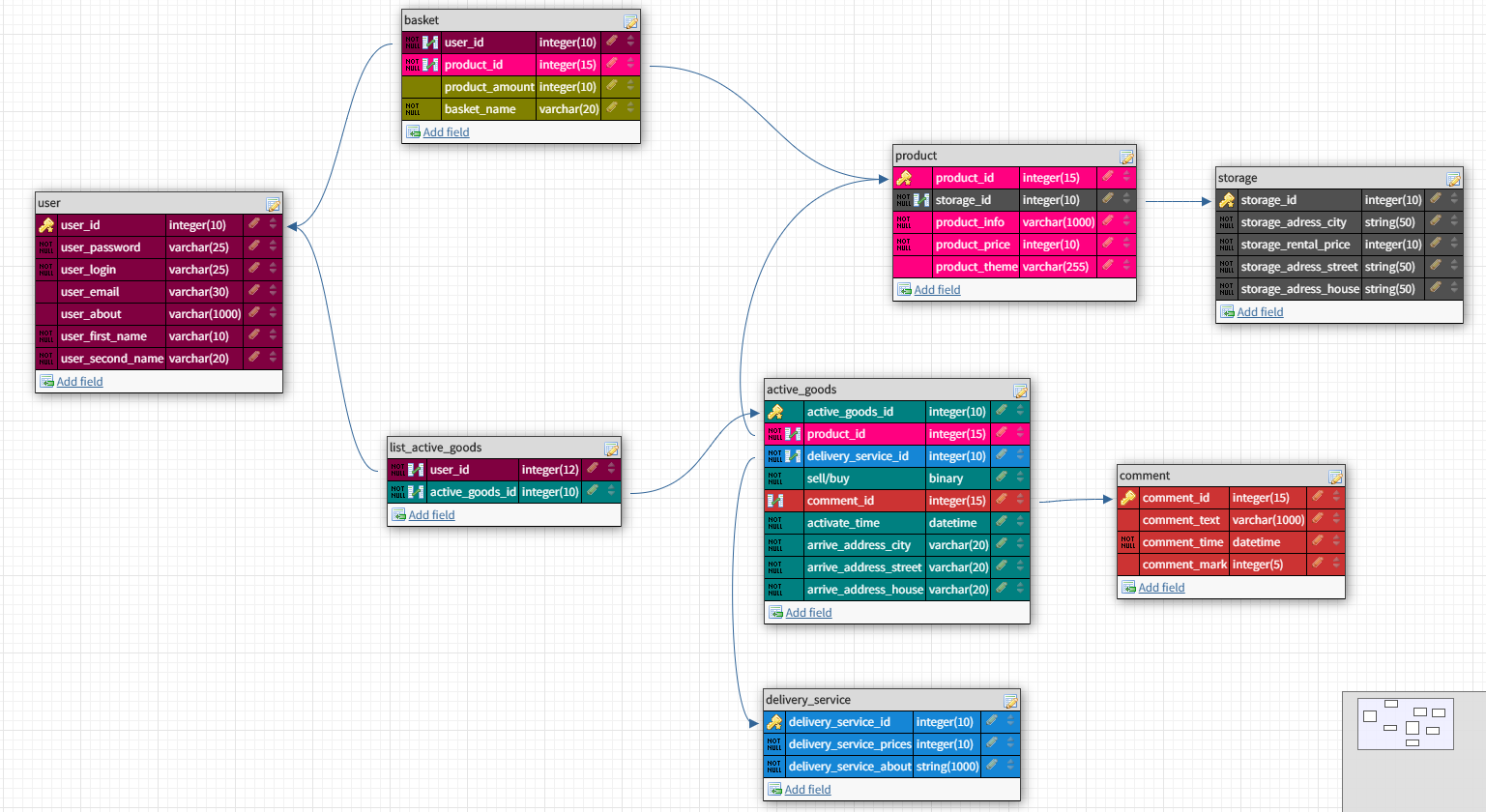
*Прийняла:*

*асистент*

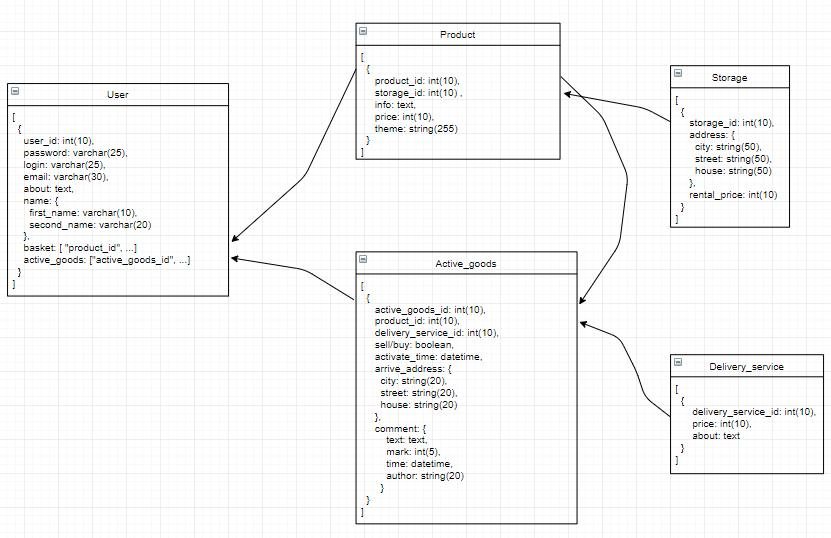
*Якимишин Х.М.*

*Львів-2020*

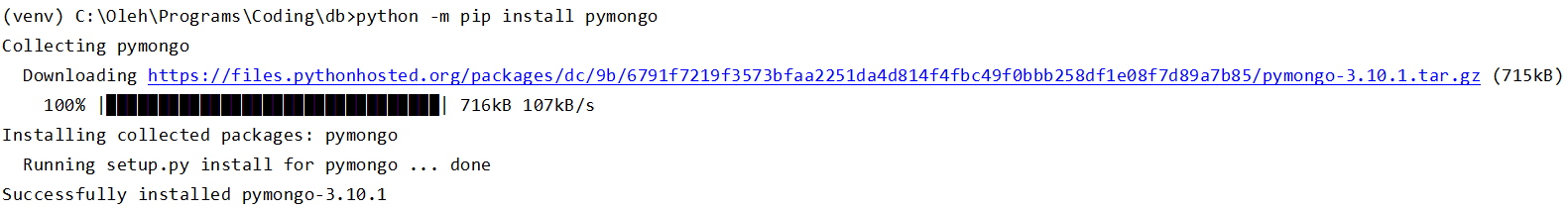
**Схема з лаб 1.**



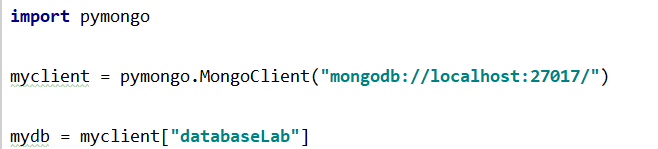
**Хід роботи.**

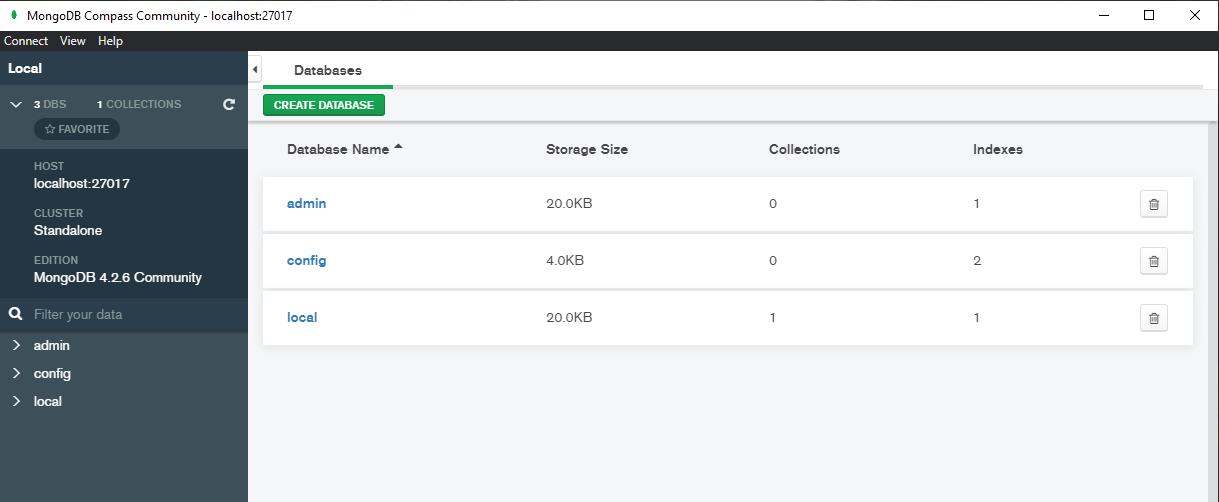


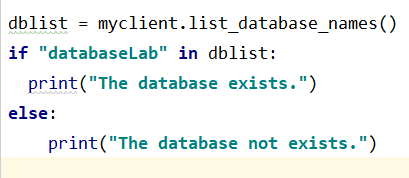
1. Для початку скачаємо драйвер для Python для роботи з mongodb



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



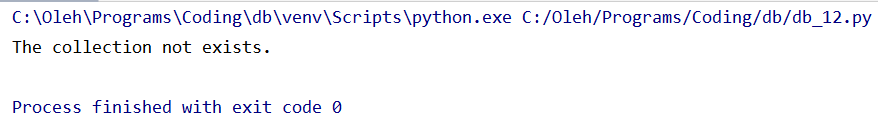
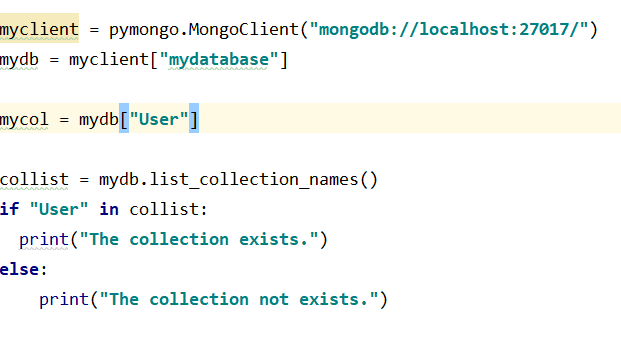






1. Створимо колекцію. Насправді вона реально створиться при заповненні першими даними

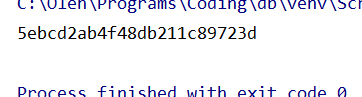




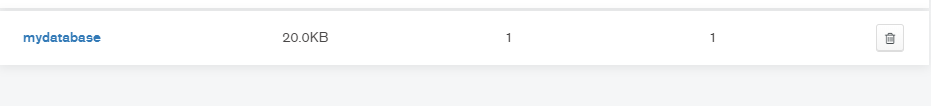
1. Заповнимо першу табличку одним рядком даних

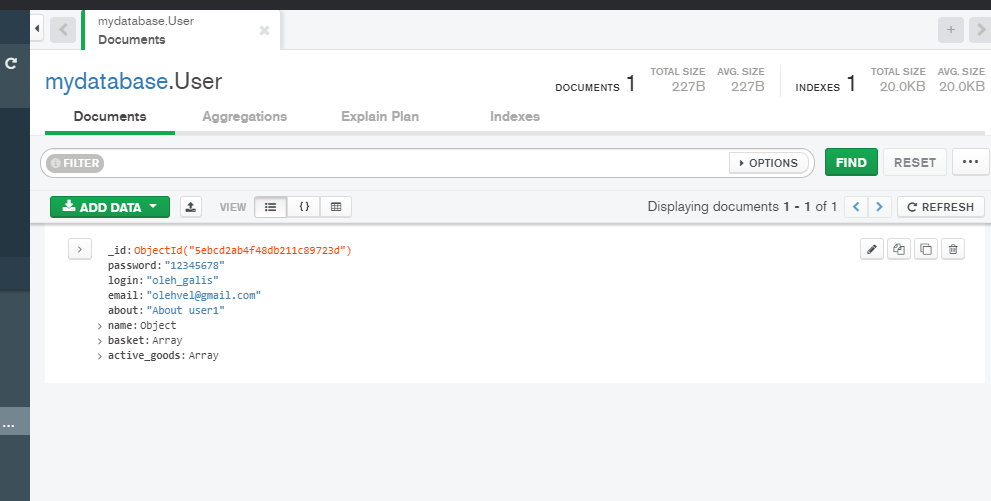
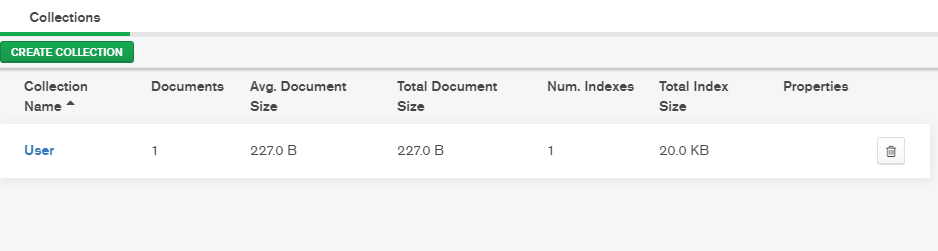


1. Виведемо \_id яке mongoDB само згенерувало



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





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

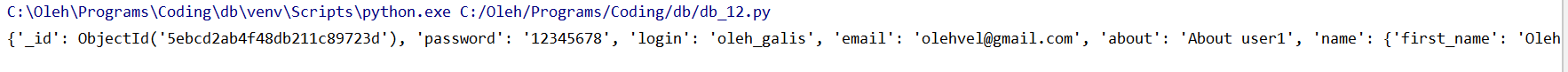
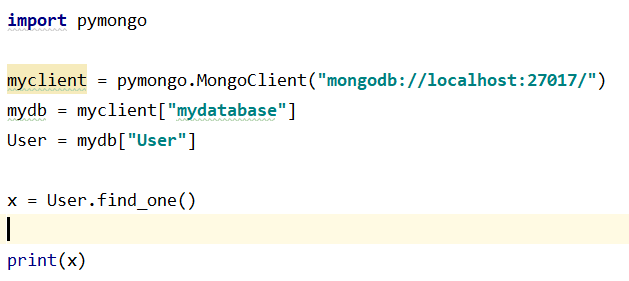
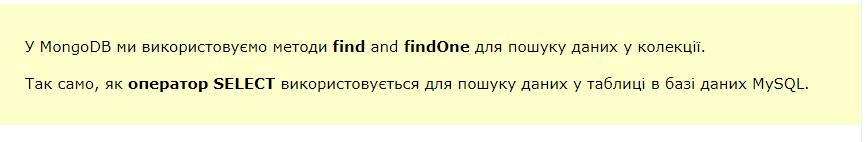
1. Введемо багато рядків у табличку.



1. Виведемо масив ід які ми задали при внесенні даних



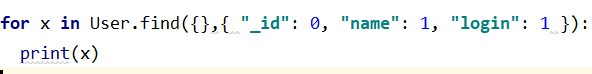
1. Спробуємо дістати дані з певної колекції. Для початку тільки один документ(перший)

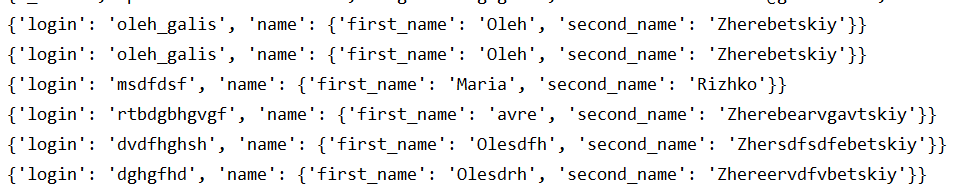


1. Тепер виведемо усі документи з колекції

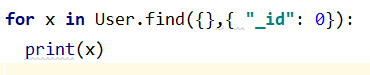


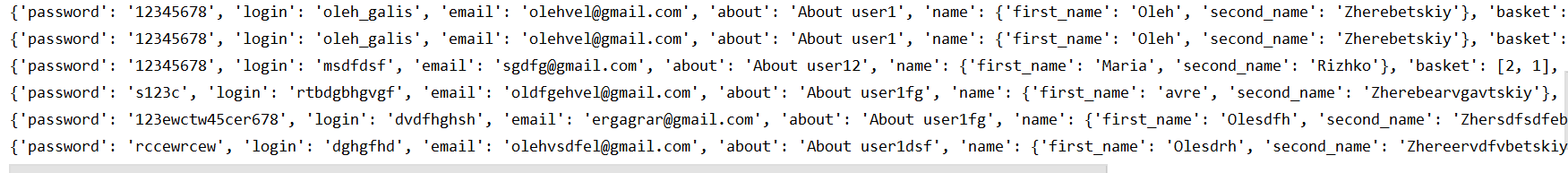
1. Задамо в пошуці поля які ми хочемо бачити(1)



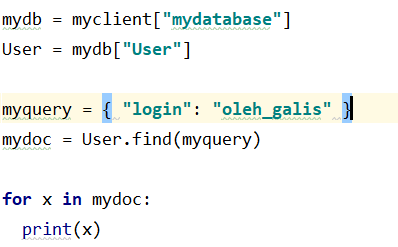


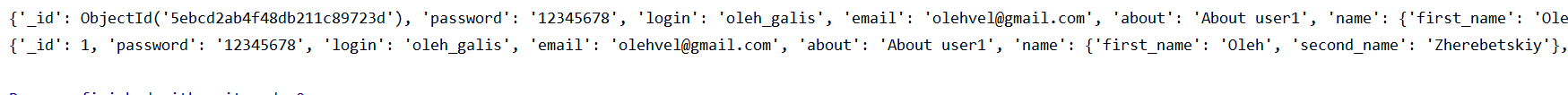
1. Виведемо все крім \_id



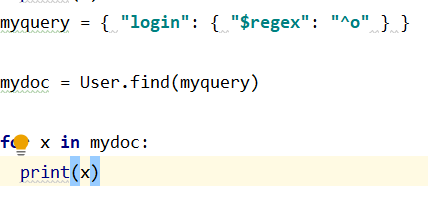


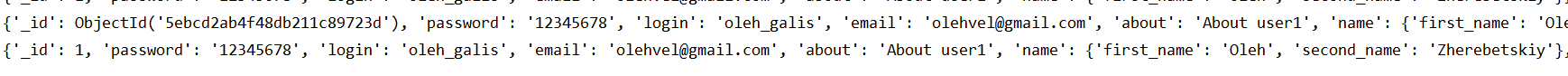
1. Проведемо при пошуці відбір тільки тих документів в яких логін саме такий…



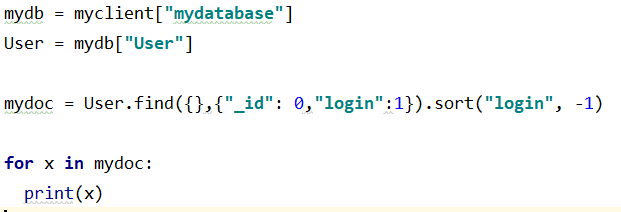


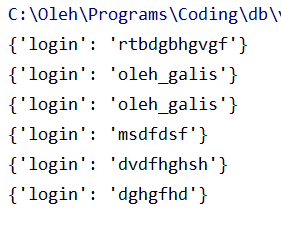
1. Використаємо регулярний вираз для відбору даних



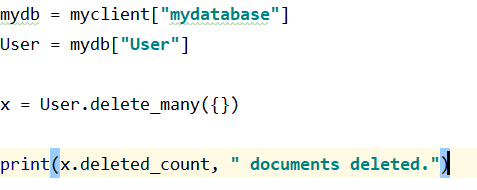


1. Виведемо тільки одне поле посортувавши по ньому по зростанню

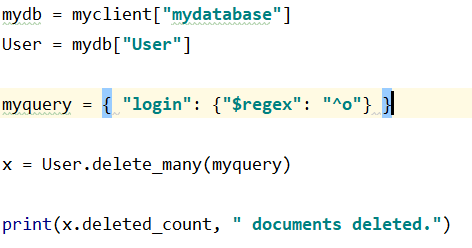




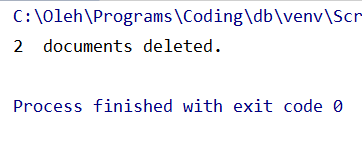
1. Видалимо всі дані з таблички



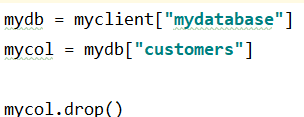
1. Видалимо тільки ті дані що відповідають певній умові



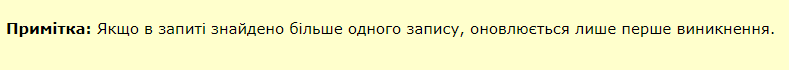
1. Виведемо скільки документів було видалено



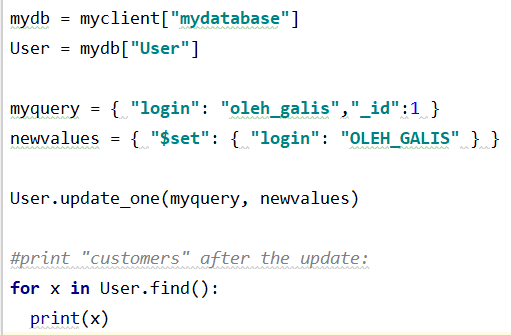
1. Видалимо колекцію



1. Оновимо Значення в одному(першому) документі певної колекції

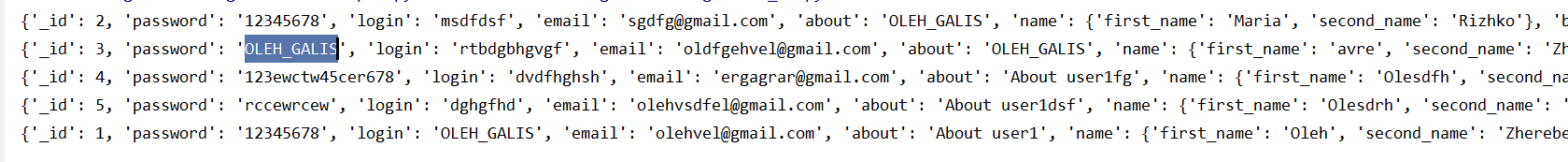


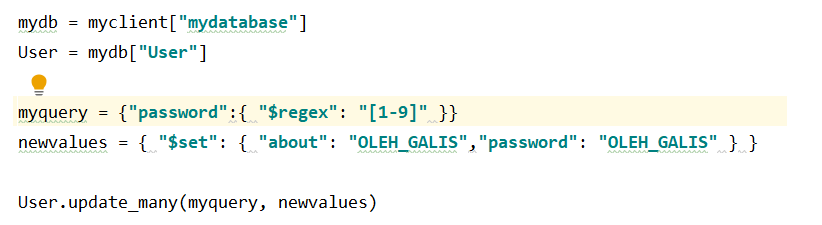




1. Оновимо дані для певних полів які підпадають під умови



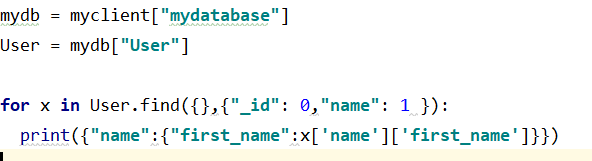


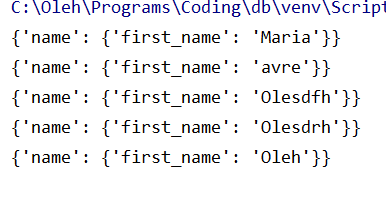




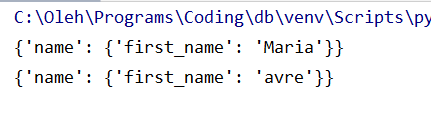
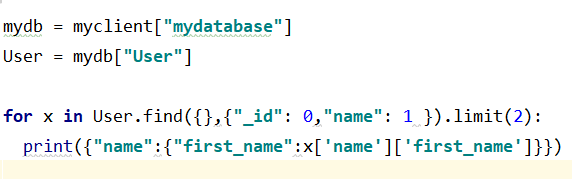


1. Якщо ми хочемо вивести певне поле об’єкта який є полем то зробимо ось так -

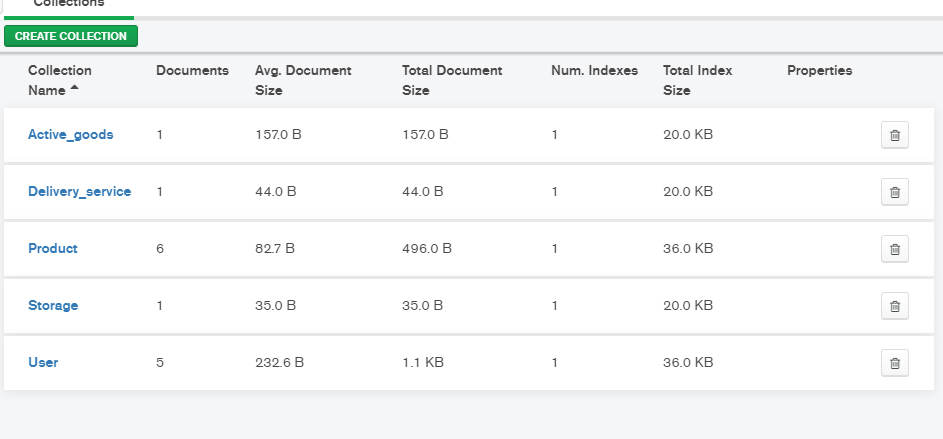
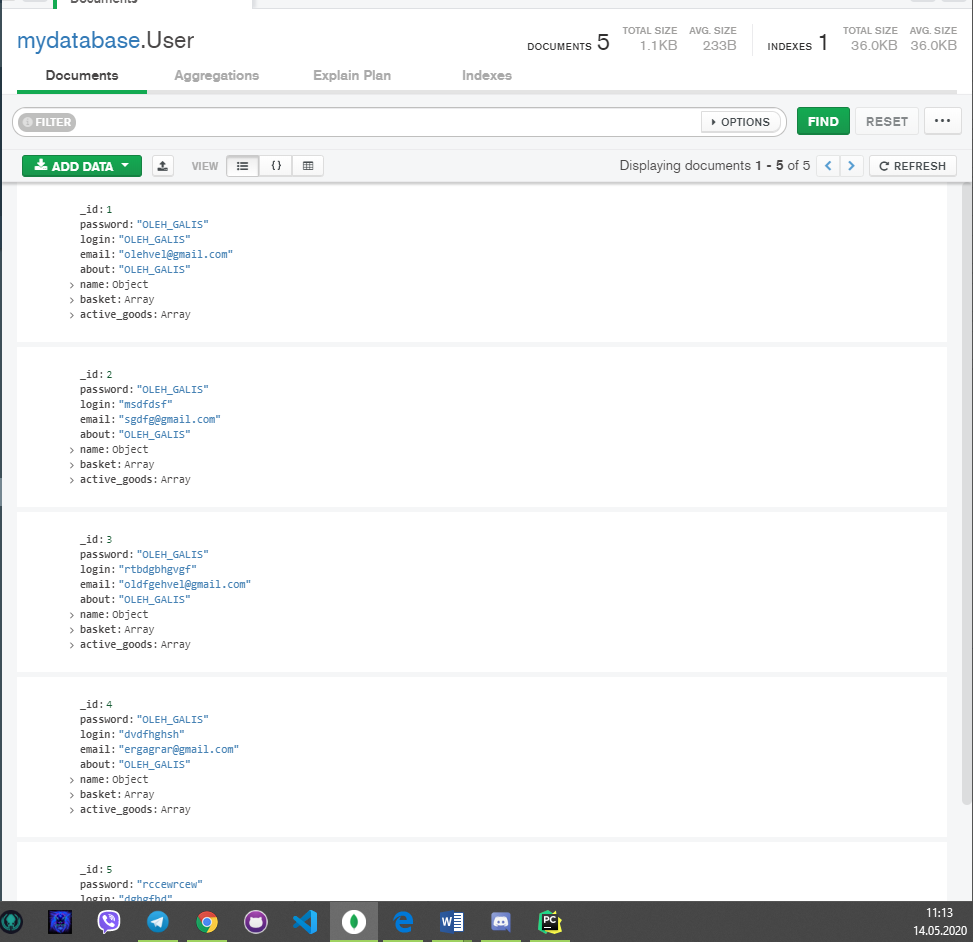




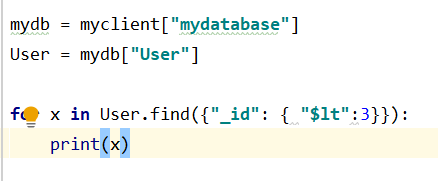
1. Обмежимо пошук лише до двох значень

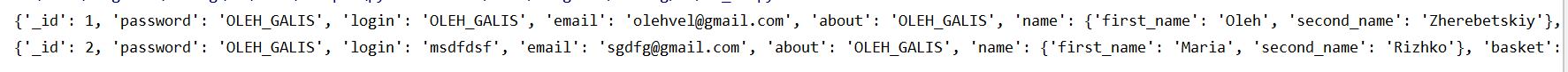


1. В результаті заглянемо в програму Compas де можемо легко побачити усі документи певних колекцій

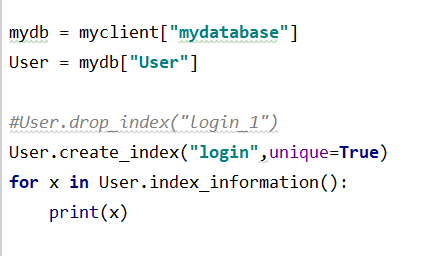


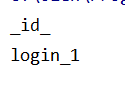
1. Використаємо умовні операції для задання умов пошуку





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





**Відповіді на питання**

1. **Назвати основні типи баз даних**

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

1. **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 – превірена часом
* SQL – доступна на більшості відомих платформах

1. **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.

1. **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 табличка
* 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"]}}})

1. **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

1. **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 елементу з кінця

1. **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 значеннь, одне з яких мусить містити поле документа
  + $nin – масив значеннь які мусить не бути в полі документа
* db.persons.find ({age: {$lt : 30}})
* db.persons.find ({age: {$gt : 30}})

1. **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

1. **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}})

1. **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.