МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ "ЛЬВІВСЬКА ПОЛІТЕХНІКА"

Кафедра систем штучного інтелекту

ЗВІТ № 14

з курсу “ОБДЗ”

на тему:

«Розробка бази даних типу NoSQL»

Виконала:

студентка групи КН-211

Лаврик Юліана

Викладач:

Якимишин Х.М.

Лабораторна робота № 14

Мета роботи: здобуття практичних навичок створення та обробки бази даних типу NoSQL на прикладі СУБД MongoDB.

Короткі теоретичні відомості

Функціональні можливості:

• узгодженість даних

• транзакції

• доступність

• можливості запитів

• масштабування

Типи значень:

• String

• Array (массив)

• Binary data (двоичные данные)

• Boolean

• Date

• Double

• Integer

• JavaScript

• Min key/Max key

• Null

• Object

• ObjectID

• Regular expression

• Symbol

• Timestamp

Операції для роботи з даними в середовищі MongoDB

Додавання даних і створення колекцій

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

> db.persons.find()

> document=({"name": "Bill", "age": "32", languages: ["english", "french"]})

> db.persons.insert(document)

Обмеження імен ключів:

Символ $ не може бути першим символом в імені ключа

Ім'я ключа не може містити символ крапки.

Ім'я _id не рекомендується використовувати

Перейменування колекції

> db.persons.renameCollection("нова_назва")

результат

{"ok" : 1}

Явне створення колекції

> db.persons.createCollection("accounts")

результат

{"ok" : 1}

Обмеження колекції

> db.createCollection("profile", {capped:true, size:9500})

{"ok":1}

> db.createCollection("profile", {capped:true, size:9500, max: 150})

Вибірка з БД

> db.persons.find()

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

Запит до вкладених об’єктів

> db.persons.insert({"name": "Alex", "age": "28", company: {"name":"microsoft", "country":"USA"}})

Налаштування запитів і сортування

> db.persons.find().limit(3)

> db.persons.find().skip(3)

> db.persons.find().sort({name: 1})

> db.persons.find().sort({name: 1}).skip(3).limit(3)

> db.persons.find({"company.name": "micriosoft"})

Команди групування

Число елементів в колекції

> db.persons.count()

> db.persons.find({name: "Tom"}).count()

> db.persons.find({name: "Tom"}).skip(2).count(true)

Функція distinct

> db.persons.distinct("name")

["Tom", "Bill", "Bob"]

Метод group

> db.persons.group ({key: {name : true}, initial: {total : 0},

reduce : function (items,prev){prev.total += 1}})

Умовні оператори

$gt (більше ніж)

$lt (менше ніж)

$gte (більше чи рівно)

$lte (менше чи рівно)

> db.persons.find ({age: {$lt : 30}})

> db.persons.find ({age: {$gt : 30}})

Оператор $ne

> db.persons.find ({age: {$ne : 22}})

Пошук по масивам і оператори $in, $nin, $all

> 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"]}})

Оператор $or

> db.persons.find ({$or : [{name: "Tom"}, {age: "22"}]})

db.ingredient.insertMany("name"="Mikl","price_for_unit":"32.25", "quantity":"30l",supplier_id:)

> db.persons.find ({name: "Tom", $or : [{age: "22"}, {languages: "german"}]})

Оператор $size

> db.persons.find ({languages: {$size:2}})

{"name": "Tom", "age": "32", languages: ["english", "german"]}

Оператор $exists

> db.persons.find ({company: {$exists:true}})

Оновлення даних

> db.persons.save({"name": "Eugene", "age" :"29", languages: ["english", "german", "spanish"]})

Функція update. прийнимає три параметра:

> 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, multi:true}) Оновлення окремого поля

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

> db.persons.update({name : "Tom"}, {$inc: {salary:100}})

Знищення поля

> db.persons.update({name : "Tom"}, {$unset: {salary: 1}})

> db.persons.update({name : "Tom"}, {$unset: {salary: 1, age: ""}})

Оператор $push

> db.persons.update({name : "Tom"}, {$push: {languages: " ukrainian "}})

Оператор $addToSet

> db.persons.update({name : "Tom"}, {$addToSet: {languages: " ukrainian "}})

> db.persons.update({name : "Tom"}, {$addToSet: {languages: {$each: ["ukrainian", "spanish", "italian"]}}})

Знищення елемента з масиву

> db.persons.update({name : "Tom"}, {$pop: {languages: 1}})

> db.persons.update({name : "Tom"}, {$pop: {languages: -1}})

> db.persons.update({name : "Tom"}, {$pull: {languages: "english"}})

> db.persons.update({name : "Tom"}, {$pullAll: {languages: ["english", "german", "french"]}})

Знищення даних

> db.persons.remove({name : "Tom"})

> db.persons.remove({name : /T\w+/i})

> db.persons.remove({age: {$lt : 30}})

> db.persons.remove({name : "Tom"}, true)

Знищення колекцій і баз даних

> db.persons.drop()

true

> db.dropDatabase()

Посилання в БД

Ручна установка посилань

> db.companies.insert({"_id" : "microsoft", "year": 1974})

> db.persons.insert({"name": "Tom", "age": 28, company: "microsoft"})

> person = db.persons.findOne()

> db.companies.findOne({_id: person.company})

Автоматичне зв'язування

> apple=({"name" : "apple", "year": 1976})

> db.companies.save(apple)

> steve = ({"name": "Steve", "age": 25, company: new DBRef('companies', apple._id)})

> db.persons.save(steve)

> db.companies.findOne({_id: steve.company.$id})

{ "$ref" : назва_колекції, "$id": значення [, "$db" : назва_бд]}

Робота з індексами

> db.persons.ensureIndex({"name" : 1})

Налаштування індексів

> db.persons.ensureIndex({"name" : 1}, {"unique" : true})

> db.persons.ensureIndex({"name" : 1, "age" : 1}, {"unique" : true})

Керування індексами

> db.system.indexes.find()

> db.persons.dropIndex("name_1")

Завдання

1. Розробити схему бази даних на основі предметної області з лабораторної роботи №1 у спосіб, що застосовується в СУБД MongoDB.
2. Забезпечити реалізацію функцій редагування, додавання та вилучення інформації в «сутність».

Хід роботи

Схема баз даних в MySQL

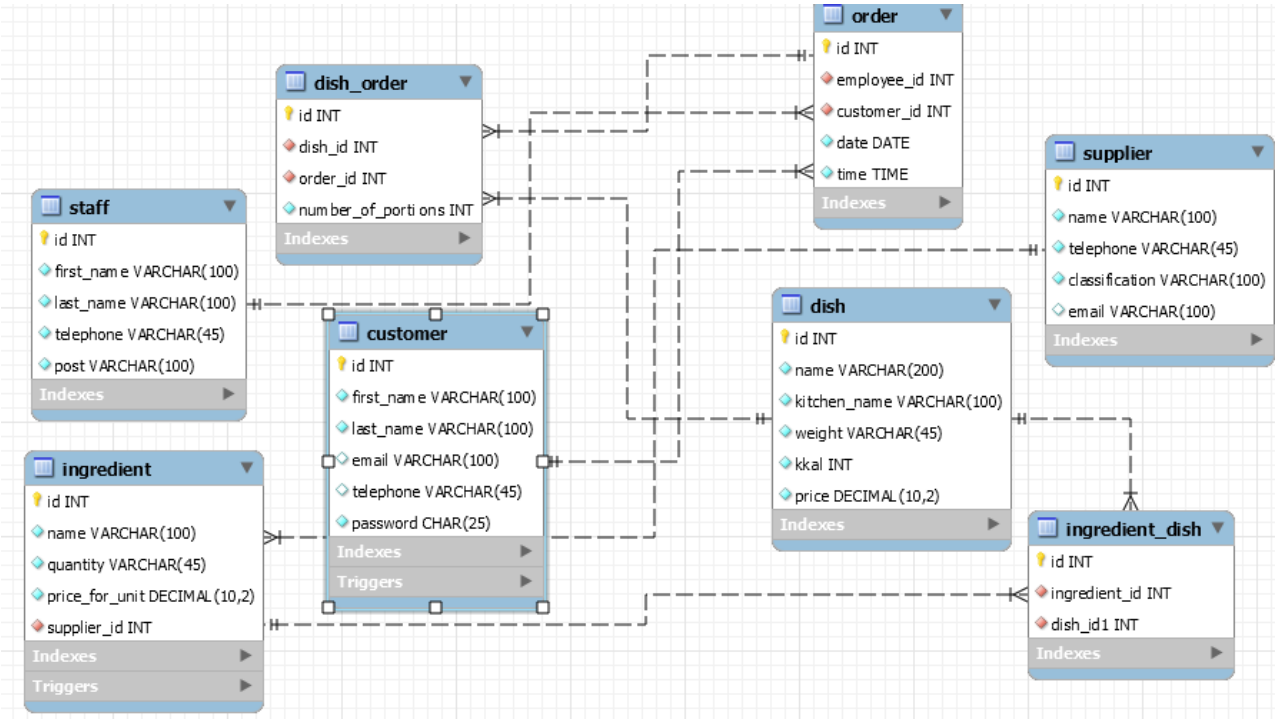

Схема баз даних в Mongodb

1. Створюємо базу даних командою *use confectionary* та відповідні колекції командою *db.createCollection("назва_колекції").*

```
> use confectionary
switched to db confectionary
> db.dropDatabase()
{ "dropped" : "confectionary", "ok" : 1 }
> use confectionary
switched to db confectionary
> db.createCollection("customer")
{ "ok" : 1 }
> db.createCollection("dish")
{ "ok" : 1 }
> db.createCollection("staff")
{ "ok" : 1 }
> db.createCollection("supplier")
{ "ok" : 1 }
> db.createCollection("ingredient")
{ "ok" : 1 }
> db.createCollection("order")
{ "ok" : 1 }
>
```

2. Перевіряємо наявні колекції за допомогою команди show collections.

```
> show collections
customer
dish
ingredient
order
staff
supplier
```

3. Заповнюємо колекцію supplier даними за допомогою команди db.назва_колекції.insertOne() та колекцію customer за допомогою команди db.назва_колекції.insertMany() . Перевіряємо чи дані внесені командою db.назва_колекції.find().

```
> db.supplier.insertOne({"name":"Molokiya","telephone":"(0352)56-12-01","classification":"dairy products"
,"email":"site@molokiya.com"})
{
        "acknowledged" : true,
        "insertedId" : ObjectId("5ec4211dd9b10d9cd425362d")
}
> db.supplier.insertOne({"name":"Hutorok","telephone":"(0352)56-13-08","classification":"floar","email":"
info@hutorok.com"})
{
        "acknowledged" : true,
        "insertedId" : ObjectId("5ec421d0d9b10d9cd425362e")
}
> db.supplier.find()
{ "_id" : ObjectId("5ec4211dd9b10d9cd425362d"), "name" : "Molokiya", "telephone" : "(0352)56-12-01", "cla
ssification" : "dairy products", "email" : "site@molokiya.com" }
{ "_id" : ObjectId("5ec421d0d9b10d9cd425362e"), "name" : "Hutorok", "telephone" : "(0352)56-13-08", "clas
sification" : "floar", "email" : "info@hutorok.com" }
```

```
> db.customer.insertMany([{"fisrt_name":"Yuliana","last_name":"Lavryk","email":"ylilav@gmail.com","telepho
ne":"+380931145078"}, {"first_name":"Anna","last_name":"Kvittkova","email":"anna12@gmail.com","telephone":
"+380675609841"}])
{
        "acknowledged" : true,
        "insertedIds" : [
                ObjectId("5ec42467d9b10d9cd425362f"),
                ObjectId("5ec42467d9b10d9cd4253630")
        ]
}
```

```
> db.customer.find()
{ "_id" : ObjectId("5ec42467d9b10d9cd425362f"), "fisrt_name" : "Yuliana", "last_name" : "Lavryk", "email" :
"ylilav@gmail.com", "telephone" : "+380931145078" }
{ "_id" : ObjectId("5ec42467d9b10d9cd4253630"), "first_name" : "Anna", "last_name" : "Kvittkova", "email" :
"anna12@gmail.com", "telephone" : "+380675609841" }
```

4. Визначаємо новий документ і вносимо його в колекцію supplier за допомогою команд назва_документа=({...}) та db.назва_колекції.save(назва_документа).

```
> roshen=({"name":"roshen","telephone":"(035212-56-87)","classification":"chocolate,cacao","email":"roshen@gmail.com"}
)
{
        "name" : "roshen",
        "telephone" : "(035212-56-87)",
        "classification" : "chocolate,cacao",
        "email" : "roshen@gmail.com"
}
> db.supplier.save(roshen)
WriteResult({ "nInserted" : 1 })
```

5. Визначаємо новий документ для таблиці ingredient та проводимо автоматичне зв’язування створеного документа та документа колекції supplier, використовуючи опцію new DBRef().

```
> cacao=({"name":"cacao", "price_for_unit":32, "quantity":"5kg","supplier_id": new DBRef('supplier',roshen._id)})
{
        "name" : "cacao",
        "price_for_unit" : 32,
        "quantity" : "5kg",
        "supplier_id" : DBRef("supplier", ObjectId("5ec429ad99a8e732c23f1f57"))
}
> db.ingredient.save(cacao)
WriteResult({ "nInserted" : 1 })
```

6. Додаємо нове поле в колекцію dish, яке буде зберігати значення індексів інгредієнтів (колекція ingredient), які входять до складу страви (колекція dish) за допомогою команди db.назва_колекції.updateOne() та заповнимо його.

За допомогою команди db.назва_колекції.find() перевіримо наявність нового поля в колекції dish.

```
>  db.dish.updateOne({}, {$set: {ingredientList: [{ingredient_id: ObjectId("5ec4339799a8e732c23f1f59") }]}}
)
{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }
> db.dish.find()
{ "_id" : ObjectId("5ec4339799a8e732c23f1f59"), "name" : "Brownie", "kitchen_name" : "American", "weight" :
 "1kg", "kkal" : 466, "price" : 455, "ingredientList" : [ { "ingredient_id" : ObjectId("5ec4339799a8e732c23
f1f59") } ] }
```

7. Додаємо дані в інші колекції .

Колекція dish:

```
> Lemon_Pie=({"name":"Lemon_Pie","kitchen_name":"England","weight":"1,2kg","kkal": 340, "price":"398"})
{
        "name" : "Lemon_Pie",
        "kitchen_name" : "England",
        "weight" : "1,2kg",
        "kkal" : 340,
        "price" : "398"
}
> db.dish.insert(Lemon_Pie)
WriteResult({ "nInserted" : 1 })
> db.dish.find()
{ "_id" : ObjectId("5ec4339799a8e732c23f1f59"), "name" : "Brownie", "kitchen_name" : "American", "weight" : "1kg",
"kkal" : 466, "price" : 455, "ingredientList" : [ { "ingredient_id" : ObjectId("5ec4339799a8e732c23f1f59") } ] }
{ "_id" : ObjectId("5ec458f1230dcfbbab66c181"), "name" : "Lemon_Pie", "kitchen_name" : "England", "weight" : "1,2kg
", "kkal" : 340, "price" : "398" }
```

Колекція staff:

```
> staff1=({"first_name":"Olena","last_name":"Bodnar","telephone":"+380734256071","post":"Chef"})
{
        "first_name" : "Olena",
        "last_name" : "Bodnar",
        "telephone" : "+380734256071",
        "post" : "Chef"
}
> db.staff.insert(staff1)
WriteResult({ "nInserted" : 1 })
> staff2=({"first_name":"Andriy","last_name":"Lavryk","telephone":"+380931256071","post":"Cook"})
{
        "first_name" : "Andriy",
        "last_name" : "Lavryk",
        "telephone" : "+380931256071",
        "post" : "Cook"
}
> db.staff.insert(staff2)
WriteResult({ "nInserted" : 1 })
> order1=({staff_id:new DBRef('staff',staff1._id),customer_id: ObjectId("5ec42467d9b10d9cd425362f"),"date_time": ne
w Date('2020-05-18T12:03:18Z'), "dishList":[{ "dish_id": Lemon_Pie._id}]})
{
        "staff_id" : DBRef("staff", undefined),
```

Колекція order:

```
> order01=({staff_id:new DBRef('staff',staff1._id),customer_id: ObjectId("5ec42467d9b10d9cd425362f"),"date_time": n
ew Date('2020-05-18T12:03:18Z'), "dishList":[{ "dish_id": ObjectId("5ec458f1230dcfbbab66c181")}]})
{
        "staff_id" : DBRef("staff", undefined),
        "customer_id" : ObjectId("5ec42467d9b10d9cd425362f"),
        "date_time" : ISODate("2020-05-18T12:03:18Z"),
        "dishList" : [
                {
                        "dish_id" : ObjectId("5ec458f1230dcfbbab66c181")
                }
        ]
}
> db.order.insert(order01)
```

8. Оновимо поле first_name в колекції staff командою db.назва_колекції.updateOne() та перевіримо виконання оновлення за допомогою команди db.назва_колекції.findOne().

```
> db.staff.updateOne({first_name: 'Olena'}, {$set: {first_name: 'Iryna'}})
{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }
> db.staff.findOne()
{
        "_id" : ObjectId("5ec45afc230dcfbbab66c182"),
        "first_name" : "Iryna",
        "last_name" : "Bodnar",
        "telephone" : "+380734256071",
        "post" : "Chef"
}
```

9. Додамо нове поле до колекції staff .

```
db.staff.updateOne({first_name: 'Andriy'}, {$set: {city: 'Lviv'}})
"acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }
```

Перевіряємо :

```
> db.staff.find({city:"Lviv"})
{ "_id" : ObjectId("5ec45b8a230dcfbbab66c183"), "first_name" : "Andriy", "last_name" : "Lavryk", "telephone" : "+38
0931256071", "post" : "Cook", "city" : "Lviv" }
```

10. Видалимо створене поле з колекції staff та зробимо перевірку.

```
> db.staff.updateOne({first_name: 'Andriy'}, {$unset: {city: 'Lviv'}})
{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }
> db.staff.find({first_name:"Andriy"})
{ "_id" : ObjectId("5ec45b8a230dcfbbab66c183"), "first_name" : "Andriy", "last_name" : "Lavryk", "telephone" : "+38
0931256071", "post" : "Cook" }
```

Контрольні запитання

1. Назвати основні типи баз даних NoSQL.
   - Бази даних виду «ключ-значення»;
   - документоорієнтовані бази даних;
   - графові бази даних;
   - колоноподібні бази даних.
2. Назвати переваги та недоліки використання баз даних NoSQL.

Переваги:

- простота роботи;

- простіший синтаксис запитів;
- кожен документ може мати власну структуру;
- можна додавати нові поля під час роботи з даними;

Недоліки:

- Обмежена ємність вбудованої мови запитів;
- Низька цінність і вузькопрофільність знань;
- Труднощі швидкого переходу з однієї нереляційної бази даних в іншу;
- Обмежена ємність вбудованої мови запитів.

3. Надати характеристику СУБД MongoDB.

MongoDB — документо-орієнтована система керування базами даних (СКБД) з відкритим вихідним кодом, яка не потребує опису схеми таблиць. MongoDB займає нішу між швидкими і масштабованими системами, що оперують даними у форматі ключ/значення, і реляційними СКБД, функціональними і зручними у формуванні запитів.

4. Операції вставки даних. insert() або save()

5. Операції оновлення даних. update() , save()

6. Операції знищення даних. update(), remove()

7. Умовні оператори.

- $gt (більше ніж)
- $lt (менше ніж)
- $gte (більше чи рівно)
- $lte (менше чи рівно)

8. Операції керування індексами. db.system.indexes.find() db.name.dropIndex("name")

9. Пошук даних. find(), findOne(), findAndModify()

10.Можливості документних БД.

- Документо-орієнтоване сховище
- Досить гнучка мова для формування запитів
- Динамічні запити

- Повна підтримка індексів
- Профілювання запитів
- Швидкі оновлення
- Ефективне зберігання даних великих обсягів, наприклад, фото та відео
- Журналювання операцій, що модифікують дані в БД

Висновок: під час виконання даної лабораторної роботи я здобула практичні навички створення та редагування бази даних типу NoSQL на прикладі СУБД MongoDB. Розробила схему бази даних на основі предметної області з лабораторної роботи №1 у спосіб, що застосовується в СУБД MongoDB, і забезпечила реалізацію функцій редагування, додавання та вилучення інформації в «сутність».