

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
«ЛЬВІВСЬКА ПОЛІТЕХНІКА»

Інститут ІКНІ

Кафедра Систем Штучного Інтелекту



Організація баз даних та знань

Лабораторна робота

Виконав:

Кіндрат В.Р.

КН-209

Викладач:

Мельникова Н. І.

Лабораторна робота №14

з курсу “ОБДЗ”

на тему: “Розробка бази даних типу NoSQL”

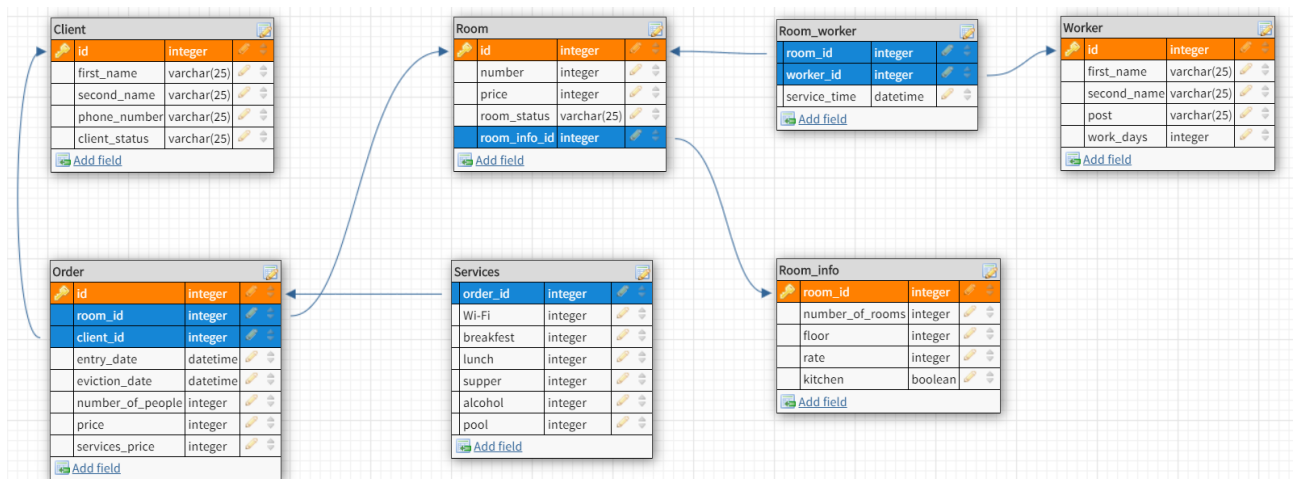
Мета роботи: здобуття практичних навичок створення та обробки бази даних типу NoSQL на прикладі СУБД MongoDB.

Хід роботи.

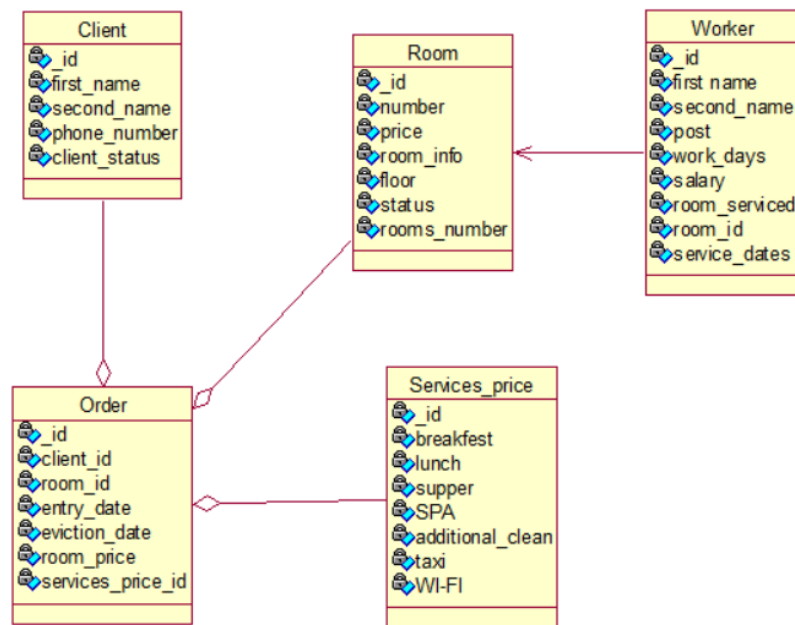
1. Розробити схему бази даних на основі предметної області з лабораторної роботи №1 у спосіб, що застосовується в СУБД MongoDB.

При проектуванні бази даних враховую особливості проектування нереляційної моделі даних та функції для роботи з БД на основі проекту готельного бізнесу. Так, як в NoSQL не реалізовані JOIN запити, а вони виконуються за сторони API немає потреби в зберіганні даних в декількох таблицях і доцільніше буде згрупувати їх. Так маючи 7 таблиць в реляційній моделі ми отримаємо 5 в нереляційній: не потрібно створювати окрему таблицю для додаткової інформації про номери готелю, також додаткова таблиця room_worker для реалізації зв'язку багато до багато не потрібна, адже в MongoDB ми можемо створювати масиви даних з посиланнями на інші таблиці. Таблицю services не змінюємо, тому що в потрібно враховувати максимальний розмір документів (16MB), одже доцільніше буде використовувати посилання на таблицю services, а не вкладати її в таблицю Order для економії пам'яті в документі.

Реляційна модель даних для готельного бізнесу:



Нереляційна модель для готельного бізнесу представлена в вигляді діаграми класів








2. Перетворити сутності діаграми БД, розробленої для лабораторної роботи №1, у структури, прийнятні для обробки в MongoDB.

Створюю потрібні колекції:

```
> use hotel
switched to db hotel
> db.createCollection("Order")
{ "ok" : 1 }
> db.createCollection("Room")
{ "ok" : 1 }
> db.createCollection("Worker")
{ "ok" : 1 }
> db.createCollection("Services")
{ "ok" : 1 }
> db.createCollection("Client")
{
  "ok" : 0,
  "errmsg" : "a collection 'hotel.Client' already exists",
  "code" : 48,
  "codeName" : "NamespaceExists"
}
```

Результата роботи функцій:

Collections						
CREATE COLLECTION						
Collection Name	Documents	Avg. Document Size	Total Document Size	Num. Indexes	Total Index Size	Properties
Order	0	-	0.0 B	1	4.1 KB	
Room	0	-	0.0 B	1	4.1 KB	
Services_price	0	-	0.0 B	1	4.1 KB	
Worker	0	-	0.0 B	1	4.1 KB	
client	1	127.0 B	127.0 B	1	20.5 KB	

3. Забезпечити реалізацію функцій редагування, додавання та вилучення інформації в «сутність».

- **Додавання інформації:**

В колекцію client:

```
db.client.insertOne({
  "_id": 1,
  "first_name": "Volodymyr",
  "second_name": "Kindrat",
  "phone_number": "38098748394",
  "client_status": "VIP"
})
```

Результат роботи функції:

```
_id: 1
first_name: "Volodymyr"
second_name: "Kindrat"
phone_number: "38098748343"
client_status: "VIP"
```

В колекцію order:

```
db.room.insertOne({
  "_id": 1,
  "number": 1,
  "price": 1700,
  "room_info": {
    "floor": 1,
    "status": "president's",
    "rooms_number": 5
  }
})
```

Результат роботи функції:

```
_id: 1
number: 1
price: 1700
✓ room_info: Object
  floor: 1
  status: "president's"
  rooms_number: 5
```

В колекцію worker:

```
db.worker.insertOne({
  "_id": 1,
  "first_name": "Lubomir",
  "second_name": "Slava",
  "post": "Cleaner",
  "work_days": 13,
  "salary": 14050,
  "room_servised": [{
    "room_id": 1,
```

```

        "service_date": [new Date("20-09-13"), new Date("20-09-17"), new
Date(" 20-09-21") ]
    },
    {"room_id": 2
        "service_date": [new Date("20-09-23"), new Date(" 20-09-29"), new
Date("20-09-31") ]
    },
    {"room_id": 3,
        "service_date": [new Date("20-10-18"), new Date("20-10-20"), new
Date("20-10-21") ]
    }
}
})

```

Результат роботи функції:

```

>
{
  _id: 1
  first_name: "Lubomir"
  second_name: "Slava"
  post: "cleaner"
  work_days: 13
  salary: 14050
  room_serviced: Array
    0: Object
      room_id: 1
      service_date: Array
        0: 2020-09-13T21:00:00.000+00:00
        1: 2020-09-17T21:00:00.000+00:00
        2: 2020-09-21T21:00:00.000+00:00
    1: Object
      room_id: 2
      service_date: Array
        0: 2020-09-23T21:00:00.000+00:00
        1: 2020-09-29T21:00:00.000+00:00
        2: 1970-01-01T00:00:00.000+00:00
    2: Object
      room_id: 3
      service_date: Array
        0: 2020-10-18T21:00:00.000+00:00
        1: 2020-10-20T21:00:00.000+00:00
        2: 2020-10-21T21:00:00.000+00:00
  }
}

```

В колекцію services_price:

```
db.services_price.insertOne({  
    "_id": 1,  
    "breakfest":1450 ,  
    "lunch":0 ,  
    "supper": 380 ,  
    "SPA":2700 ,  
    "additional_clean":0 ,  
    "taxi":508 ,  
    "WI-FI":578 })
```

Результат роботи функції:

```
_id: 1  
breakfest: 1450  
lunch: 0  
supper: 380  
SPA: 2700  
additional_clean: 0  
taxi: 508  
WI-FI: 578
```

В колекцію order:

```
db.order.insertOne({  
    "_id": 1,  
    "client_id":1 ,  
    "room_id":1 ,  
    "services_price_id":1 ,  
    "entry_date":new Date("20-09-15") ,  
    "eviction_date": new Date("20-09-23 "),  
    "room_price":13450  
})
```

Результат роботи функції:

```
_id: 1
client_id: 1
room_id: 1
services_price_id: 1
entry_date: 2020-09-14T00:00:00.000+00:00
eviction_date: 1970-01-01T00:00:00.000+00:00
room_price: 13450
```

- Редагування інформації:

Змінимо дату виїзду для запису в сутності order з _id = 1

```
db.order.updateOne({_id: 1},{$set:{eviction_date: new Date("20-09-24")}})
> db.order.updateMany({_id: 1},{$set:{eviction_date: new Date("2020-09-24")}})
{"acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }
```

Результат роботи функції:

```
_id: 1
client_id: 1
room_id: 1
services_price_id: 1
entry_date: 2020-09-14T00:00:00.000+00:00
eviction_date: 2020-09-24T00:00:00.000+00:00
room_price: 13450
```

Також створи ще одну функцію, яка змінить статус всіх кімнат, в яких ціна більше 4000 за добу на "president's". Для цього створимо ще три кімнати, де 2 з них матимуть room_price більше, або рівне 4000 та status: "regular":

```
db.room.insertOne({_id:2, "numebr":9, "room_price": 4300,
"room_info":{"floor":1, "status": "regular", "rooms_naumber": 4}})
db.room.insertOne({_id:3, "numebr":5, "room_price": 1900,
"room_info":{"floor":1, "status": "regular", "rooms_naumber": 2}})
db.room.insertOne({_id:4, "numebr":11, "room_price": 5500,
"room_info":{"floor":2, "status": "regular", "rooms_naumber": 5}})
```

```
> db.room.insertOne({_id:2, "numebr":9, "price": 4300, "room_info":{"floor":1, "status": "regular", "rooms_naumber": 4}})
{"acknowledged" : true, "insertedId" : 2 }
> db.room.insertOne({_id:3, "numebr":5, "price": 1900, "room_info":{"floor":1, "status": "regular", "rooms_naumber": 2}})
{"acknowledged" : true, "insertedId" : 3 }
> db.room.insertOne({_id:4, "numebr":11, "price": 5500, "room_info":{"floor":2, "status": "regular", "rooms_naumber": 5}})
{"acknowledged" : true, "insertedId" : 4 }
```


Результат роботи функції:

```
> {
  _id: 2
  numebr: 9
  room_price: 4300
  room_info: Object
    status: "regular"
}
```

```
{
  _id: 3
  numebr: 5
  room_price: 1900
  room_info: Object
    floor: 1
    status: "regular"
    rooms_naumber: 2
}
```

```
{
  _id: 4
  numebr: 11
  room_price: 5500
  room_info: Object
    status: "regular"
}
```

Тепер створимо функцію для зміни інформації

```
db.room.updateMany({
  room_price: {$gte: 4000}},
{$set: {room_info:
  {status: "president`s"}}
})
```

```
> db.room.updateMany({room_price: {$gte: 4000}}, {$set: {room_info: {status: "president`s"}}})
{ "acknowledged" : true, "matchedCount" : 2, "modifiedCount" : 2 }
```

Результат роботи функції:

```
> {
  _id: 2
  numebr: 9
  room_price: 4300
  room_info: Object
    status: "president`s"
}
```

```
{
  _id: 3
  numebr: 5
  room_price: 1900
  room_info: Object
    floor: 1
    status: "regular"
    rooms_naumber: 2
}
```

```
_id: 4
numebr: 11
room_price: 5500
✓ room_info: Object
  status: "president`s"
```

- **Вилучення інформації:**

Створимо запис в сутності order та дамо йому client_id: 2

```
db.order.insertOne({
  "_id": 2,
  "client_id": 2,
  "room_id": 1,
  "services_price_id": 2,
  "entry_date": new Date("20-09-04"),
  "eviction_date": new Date("20-09-07 "),
  "room_price": 5600
})
```

Результат роюоти функції:

```
_id: 1
client_id: 1
room_id: 1
services_price_id: 1
entry_date: 2020-09-14T00:00:00.000+00:00
eviction_date: 1970-01-01T00:00:00.000+00:00
room_price: 13450
```

```
_id: 2
client_id: 2
room_id: 1
services_price_id: 2
entry_date: 2020-09-04T00:00:00.000+00:00
eviction_date: 1970-01-07T00:00:00.000+00:00
room_price: 5600
```

Та видалимо записи в яких client_id більше 1:

```
db.order.deleteMany({client_id: {$gt: 1}})
```

```
> db.order.deleteMany({client_id:{$gt:1}})
{ "acknowledged" : true, "deletedCount" : 1 }
```

Результат роботи функції:

```
_id: 1
client_id: 1
room_id: 1
services_price_id: 1
entry_date: 2020-09-14T00:00:00.000+00:00
eviction_date: 1970-01-01T00:00:00.000+00:00
room_price: 13450
```

Видалимо всі поля з таблиці services_price, де additional_clean = 0:"

```
db.services_price.update({_id: 1},{unset:{additional_clean:0}})
```

```
> db.services_price.update({_id: 1},{unset:{additional_clean:0}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
```

Результат роботи функції:

```
_id: 1
breakfast: 1450
lunch: 0
supper: 380
SPA: 2700
taxi: 508
WI-FI: 578
```

Висновок:

Під час виконання лабораторної роботи я здобув практичних навичок створення та обробки бази даних типу NoSQL на прикладі СУБД MongoDB. Навчився проектувати не реляційні моделі даних для СУБД MongoDB.