**КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ**

**імені ТАРАСА ШЕВЧЕНКА**

****

**ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ**

**Кафедра прикладних інформаційних систем**

**Звіт до практичної роботи №5**

**з курсу**

**«Бази Даних»**

*студента 2 курсу*

*групи ПП-22*

*спеціальності 122 «Комп'ютерні науки»*

*ОП «Прикладне програмування»*

Шевлюк Вікторії Віталіївни

*Викладач:*

асистент

Криволапов Я. В.

**Київ – 2022**

**Тема:** проектування постреляційних баз даних.

**Мета:** Набуття навичок по проектуванню постреляційних баз даних. Створення власної бази даних в СУБД MongoDB. Отримання даних з використанням запитів.

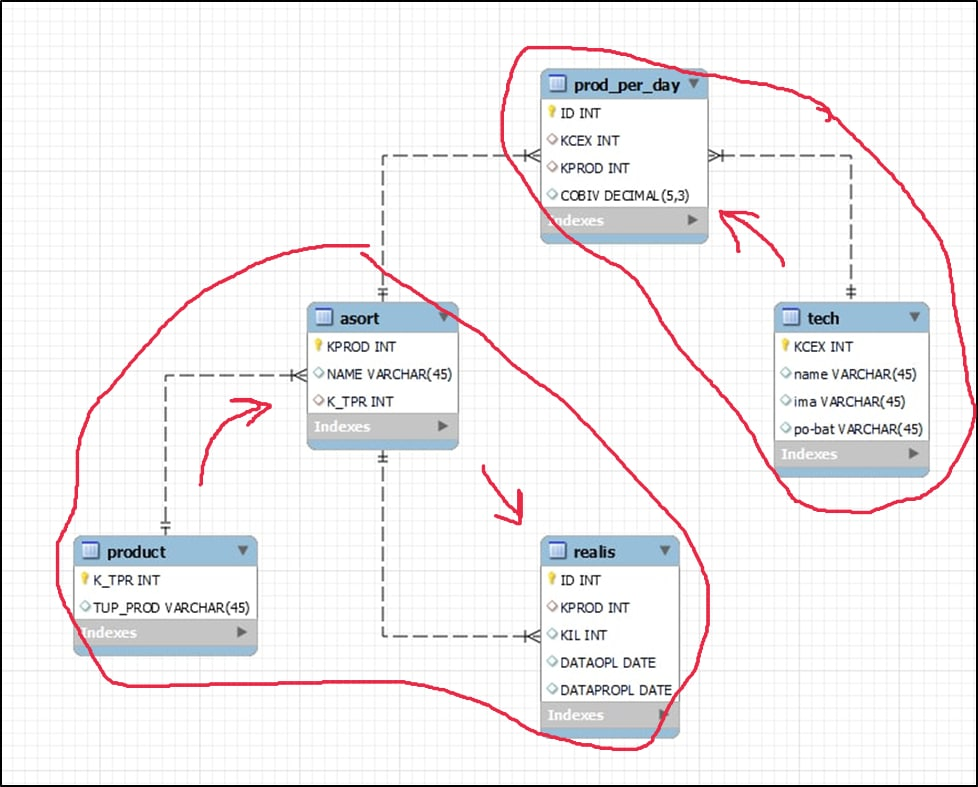
**Завдання:**

Відповідно до обраного варіанту та індивідуального завдання спроектувати схеми документів для заповнення бази даних в СУБД MongoDB на 2 колекції. Виконати відповідні запити (1.1, 1.2, 1.3) до колекцій в MongoDB.

**Хід роботи:**

Почнемо роботу зі створення схеми документів для заповнення СУБД. Це можна зробити за допомогою текстового файлу.

В моєму варіанті всього 5 таблиць, які я об’єднала у колекції наступним чином:



Ось так виглядає схема документів для колекції із трьох таблиць:

db.prodrealis.insertMany([{"product":{"KPROD": 12, "COBIV": 5.02, "NAME": "Пташка", "K\_TPR": 101, "TUP\_PROD": "Крамель"}, "KIL": 100, "DATAOPL": new Date(2005,4,11), "DATAPROPL": new Date(2005,4,12)},

{"product":{"KPROD": 15, "COBIV": 5.89, "NAME": "Молочна", "K\_TPR": 101, "TUP\_PROD": "Крамель"}, "KIL": 100, "DATAOPL": new Date(2005,4,11), "DATAPROPL": new Date(2005,4,26)},

{"product":{"KPROD": 18, "COBIV": 4.69, "NAME": "Театральна", "K\_TPR": 101, "TUP\_PROD": "Крамель"}, "KIL": 85, "DATAOPL": new Date(2005,4,15), "DATAPROPL": new Date(2005,4,20)},

{"product":{"KPROD": 26, "COBIV": 3.85, "NAME": "Мулатка", "K\_TPR": 202, "TUP\_PROD": "Цукерки"}, "KIL": 74, "DATAOPL": new Date(2005,4,19), "DATAPROPL": new Date(2005,4,24)},

{"product":{"KPROD": 25, "COBIV": 6.25, "NAME": "Хід королеви", "K\_TPR": 202, "TUP\_PROD": "Цукерки"}, "KIL": 65, "DATAOPL": new Date(2005,4,23), "DATAPROPL": new Date(2005,4,28)},

{"product":{"KPROD": 29, "COBIV": 5.89, "NAME": "Ромашка", "K\_TPR": 202, "TUP\_PROD": "Цукерки"}, "KIL": 85, "DATAOPL": new Date(2005,4,27), "DATAPROPL": new Date(0000,00,00)},

{"product":{"KPROD": 32, "COBIV": 7.01, "NAME": "Чайка", "K\_TPR": 303, "TUP\_PROD": "Шоколад"}, "KIL": 49, "DATAOPL": new Date(2005,5,01), "DATAPROPL": new Date(2005,5,06)},

{"product":{"KPROD": 33, "COBIV": 1.52, "NAME": "Оленка", "K\_TPR": 303, "TUP\_PROD": "Шоколад"}, "KIL": 100, "DATAOPL": new Date(2005,5,05), "DATAPROPL": new Date(200,5,15)},

{"product":{"KPROD": 36, "COBIV": 1.48, "NAME": "Гвардейский", "K\_TPR": 303, "TUP\_PROD": "Шоколад"}, "KIL": 125, "DATAOPL": new Date(2005,5,09), "DATAPROPL": new Date(2005,5,14)},

{"product":{"KPROD": 12, "COBIV": 5.02, "NAME": "Пташка", "K\_TPR": 101, "TUP\_PROD": "Крамель"}, "KIL": 135, "DATAOPL": new Date (2005,5,13), "DATAPROPL": new Date(2005,5,18)},

{"product":{"KPROD": 15, "COBIV": 5.89, "NAME": "Молочна", "K\_TPR": 101, "TUP\_PROD": "Крамель"}, "KIL": 98, "DATAOPL": new Date(2005,5,17), "DATAPROPL": new Date(2005,5,22)},

{"product":{"KPROD": 18, "COBIV": 4.69, "NAME": "Театральна", "K\_TPR": 101, "TUP\_PROD": "Крамель"}, "KIL": 76, "DATAOPL": new Date(2005,5,21), "DATAPROPL": new Date(2005,5,26)},

{"product":{"KPROD": 26, "COBIV": 3.85, "NAME": "Мулатка", "K\_TPR": 202, "TUP\_PROD": "Цукерки"}, "KIL": 102, "DATAOPL": new Date(2005,5,25), "DATAPROPL": new Date(2005,5,30)},

{"product":{"KPROD": 25, "COBIV": 6.25, "NAME": "Хід королеви", "K\_TPR": 202, "TUP\_PROD": "Цукерки"}, "KIL": 165, "DATAOPL": new Date(2005,5,29), "DATAPROPL": new Date(2005,6,07)},

{"product":{"KPROD": 29, "COBIV": 5.89, "NAME": "Ромашка", "K\_TPR": 202, "TUP\_PROD": "Цукерки"}, "KIL": 102, "DATAOPL": new Date(2005,6,02), "DATAPROPL": new Date(0000,00,00)}])

І ось так виглядає схема документів з двох таблиць:

db.proddetails.insertMany([{"Вироблено": {"KCEX": 1, "name": "Проценко", "ima": "Іван", "pobat": "Семенович"}, "KPROD": 12, "COBIV": 5.02},

{"Вироблено": {"KCEX": 1, "name": "Проценко", "ima": "Іван", "pobat": "Семенович"}, "KPROD": 15, "COBIV": 5.89},

{"Вироблено": {"KCEX": 1, "name": "Проценко", "ima": "Іван", "pobat": "Семенович"}, "KPROD": 18, "COBIV": 4.69},

{"Вироблено": {"KCEX": 2, "name": "Кравченко", "ima": "Кирило", "pobat": "Сергійович"}, "KPROD": 26, "COBIV": 3.85},

{"Вироблено": {"KCEX": 2, "name": "Кравченко", "ima": "Кирило", "pobat": "Сергійович"}, "KPROD": 25, "COBIV": 6.25},

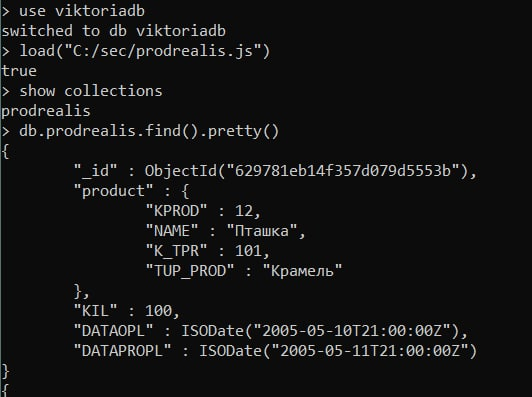
{"Вироблено": {"KCEX": 2, "name": "Кравченко", "ima": "Кирило", "pobat": "Сергійович"}, "KPROD": 29, "COBIV": 5.89},

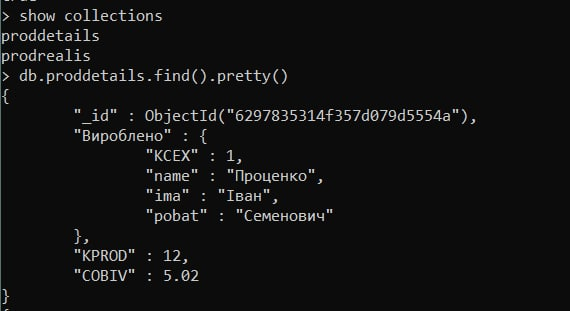
{"Вироблено": {"KCEX": 3, "name": "Жовніров", "ima": "Юрій ", "pobat": "Петрович"}, "KPROD": 32, "COBIV": 7.01},

{"Вироблено": {"KCEX": 3, "name": "Жовніров", "ima": "Юрій ", "pobat": "Петрович"}, "KPROD": 33, "COBIV": 1.52},

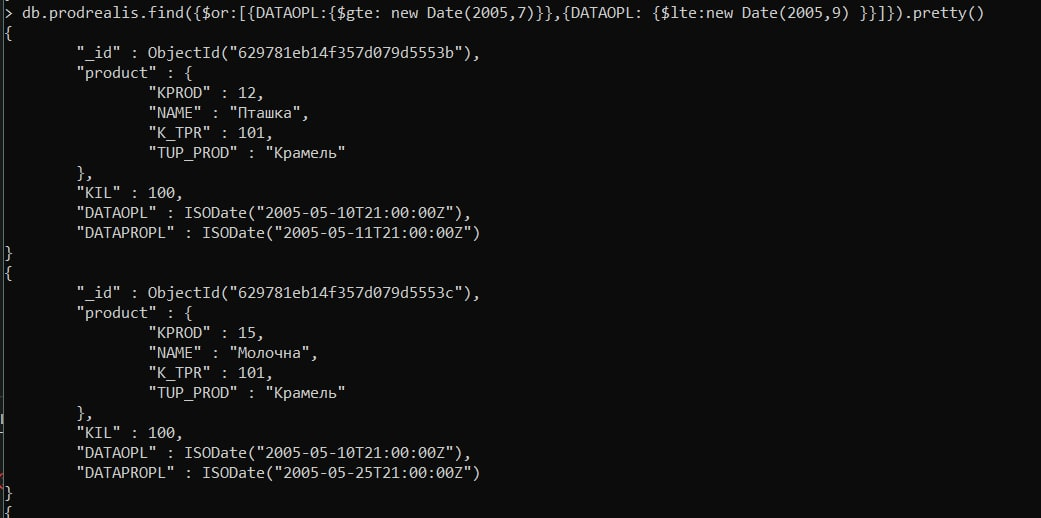
{"Вироблено": {"KCEX": 3, "name": "Жовніров", "ima": "Юрій ", "pobat": "Петрович"}, "KPROD": 36, "COBIV": 1.48}])

Щоб завантажити наші колекції, заходимо в MongoDB і вводимо команду **use viktoriadb** щоб створити нову базу даних та перейти в неї. Коли ми опинились в потрібній базі, підвантажуємо туди колекції за допомогою команди **load().** В параметрах ціїє команти потрібно прописати шлях до текстових файлів, наведених вищу. До речі, для коректності роботи цієї функції я зберігла текстові файли з розширенням **js.**





За допомогою команди db.collection.find().pretty() можна отримати акуратний вивід даних з таблиці:



Далі, коли дані додані до колекцій, ми можемо виконувати запити.

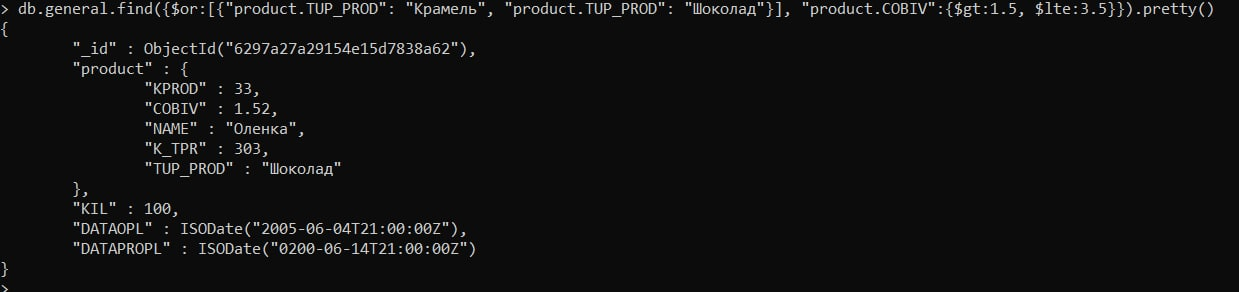
**Запит 1.1**

Інформацію про виробництво шоколадних і карамельних виробів, собівартість яких лежить в межах [1,5; 3,5] грн.

Код запиту:

db.general.find({$or:[{"product.TUP\_PROD": "Крамель", "product.TUP\_PROD": "Шоколад"}], "product.COBIV":{$gt:1.5, $lte:3.5}}).pretty()

Якщо переглянути в колекцї дані, що підходять ід цей запит – має вивести лише один об’єкт. Переглянемо результат запиту:



Як бачимо, запит працює коректно.

Перейдемо до наступного.

**Запит 1.2**

Інформацію про продукцію кондитерської фабрики, що була реалізована у третьому кварталі 2005 року.

Даний запит я виконала двома способами. За допомогою команди **find():**

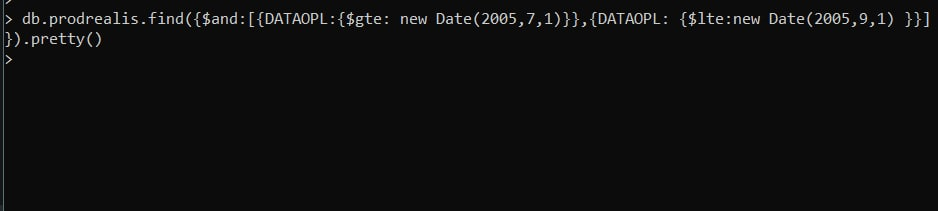
> db.prodrealis.find({$and:[{DATAOPL:{$gte: new Date(2005,7,1)}},{DATAOPL: {$lte:new Date(2005,9,1) }}]}).pretty()

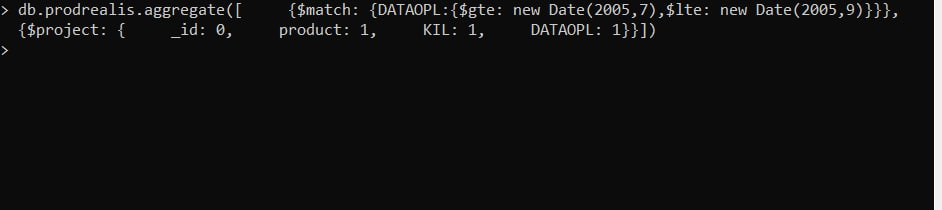
І за допомогою **aggregate():**

db.prodrealis.aggregate([ {$match: {DATAOPL:{$gte: new Date(2005,7),$lte: new Date(2005,9)}}}, {$project: { \_id: 0, product: 1, KIL: 1, DATAOPL: 1}}])

Якщо переглянути колекції, то там немає даних, які б задовольняли умови цих запитів. Отже, ми маємо отримати порожній результат.

Переглянемо результати що ми отримали:





Як бачимо, запити працюють коректно.

Далі останній запит, його я також виконала за допомогою **aggregate():**

**Запит 1.3**

Дані про продукцію, яке має найбільшу реалізацію (кількість записів про найбільшу реалізацію не повинна перевищувати восьми). Динамічний набір записів складається з таких полів: *Код продукції, Повна назва продукції, Кількість продукції.*

db.prodrealis.aggregate([

{$sort: {KIL: -1 } },

{$limit: 8},

{$project: {

\_id: 0,

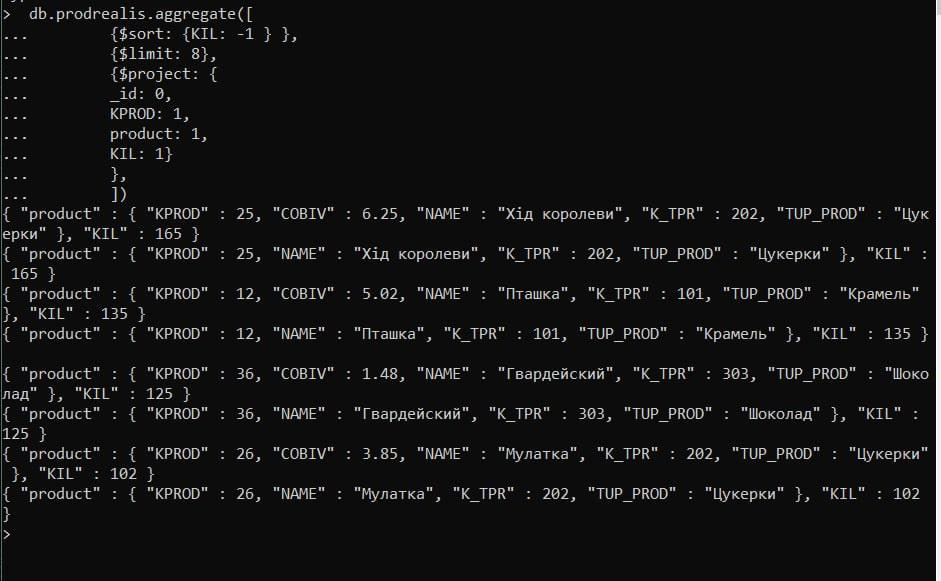
KPROD: 1,

product: 1,

KIL: 1}

},

])



Запит видає нам поля з найбільшою реалізацією, і цих полів рівно 8. Запит працює корректно.

**Висновок:** У ході виконання лабораторної роботи я набула навичок по проектуванню постреляційних баз даних, створенню власної бази даних в СУБД MongoDB та отриманню даних з використанням запитів.

**Контрольні запитання:**

1. **Основні характеристики СУБД MongoDB.**

Документо-орієнтоване сховище (проста та потужна JSON-подібна схема даних), гнучка мова для формування запитів, динамічні запити, повна підтримка індексів, профілювання запитів, швидкі оновлення «на місці», ефективне зберігання бінарних даних великих обсягів, журналювання операцій, що модифікують дані в БД, підтримка відмовостійкості і масштабованості: асинхронна реплікація, набір реплік і шардінг, може працювати відповідно до парадигми MapReduce

1. **Операції CRUD в MongoDB.**

► Create (створення db.collection.insertMany()),

► Read (пошук або читання db.collection.find()),

► Update (оновлення db.collection.updateOne(), db.collection.updateMany()),

► Delete (видалення db.collection.deleteMany()).

1. **Значення команди find().**

Вибирає документи в колекції або подання та повертає курсор до вибраних документів.

1. **Правила формування складних запитів.**

Для формування складного запиту потрібно написати назву колекції, функцію та параметри даної функції. В якості параметрів можуть бути якісь ключі в колекції та певні дії з ними для формування запиту.

1. **Побудова індексів та команда explain().**

У старших чотирьох байтах знаходиться стандартна часова мітка Unix, що містить кількість секунд від початку "епохи". Наступні три байта - ідентифікатор комп’ютера, а наступні два - ідентифікатор процесу. Останні три байта - локальний для процесу лічильник, який збільшується при генерації кожного наступного ідентифікатора об’єкта.

Команада explain() надає інформацію про результати виконання певиних команд, наприклад: aggregate, count, distinct, find, findAndModify, delete, mapReduce, та update.

1. **Отримання довідки в MongoDB.**

db.help()

1. **Принципи проектування схеми бази даних в MongoDB.**

Схема будується на основі того, що з базою даних будуть робити, адже MongoDb дає можливість створення колекції з даними з декількох таблиці.

Сама побудова складається з етапів:

► системний аналіз предметної області;

► концептуальне проектування;

► логічне проектування;

► фізичне проектування.

1. **Принципи встановлення ключових полів в MongoDB.**

В якості ключових полів у MongoDb виступають індекси, що розставляються автоматично. Дані індекси є універсальними, оскільки в одній колекції можуть зберігатись різні типи даних.