МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО”

Факультет прикладної математики

Кафедра програмного забезпечення комп’ютерних систем

КУРСОВИЙ ПРОЕКТ

з дисципліни “Бази даних”

спеціальність 121 – Програмна інженерія

на тему: Моніторингова система ресурсів з продажами смартфонів

Студента

групи КП-82 Суходольський Є.В. _____

(підпис)

Викладач

к.т.н, доцент кафедри

СПіСКС **Петрашенко А.В.** _____

(підпис)

Захищено з оцінкою _____

Київ – 2021

Анотація

Метою розробки даного курсового проекту є набуття практичних навичок розробки сучасного програмного забезпечення, що взаємодіє з постреляційними базами даних, а також здобуття навичок оформлення відповідного текстового, програмного та ілюстративного матеріалу у формі проектної документації. У результаті виконання курсового проекту було опановано навички розроблення програмного забезпечення для постреляційних баз даних, володіння основами використання СУБД, а також інструментальними засобами аналізу великих обсягів даних.

Темою даного курсового проекту є створення моніторингової системи ресурсів з продажами смартфонів. У документі викладена актуальність та проблематика аналізу великого обсягу даних, аналіз використаного інструментарію (опис мови програмування, використаних бібліотек та СУБД), описана структура бази даних, опис розробленого програмного забезпечення (загальний, опис модулів та основних алгоритмів роботи), аналіз функціонування засобів масштабування, та опис результатів проведеного аналізу.

Результатами даного проекту стали діаграми та графіки, що зображають результати аналізу смартфонів та їх основних характеристик. З ними можна ознайомитися в додатку А.

Зміст

Вступ

Data Science – галузь інформатики, що вивчає проблеми аналізу, обробки і представлення даних у цифровій формі. Говорячи простіше, це наука про методи обробки великих масивів даних і вилучення з них цінної інформації, завдяки чому можна більш ефективно приймати рішення. Все це стало можливо завдяки появі хмарних сервісів для зберігання даних, зростання обчислювальних здібностей комп'ютерів, розвитку технологій машинного навчання і нейромереж.

Сучасний Data Science спеціаліст (дослідник даних) має оперувати великими обсягами даних та вміти виокремити з них приховані залежності, на основі яких зробити прогноз про те, як будуть надалі відбуватися ті чи інші явища. Дослідники даних використовують свої дані та аналітичні здібності для пошуку та інтерпретації великих джерел даних; керують великими обсягами даних безвідносно до апаратного та програмного забезпечення і обмежень пропускної здатності; об'єднують джерела даних; забезпечують цілісність наборів даних; створюють візуалізації для кращого розуміння даних; з використанням даних будують математичні моделі; надають тлумачення даних та висновки.

Метою даного курсового проекту був аналіз смартфонів за допомогою моніторингової системи. Мета дослідження полягає в прогнозуванні ціни на смартфони в залежності від кількості ядер процесора, тактової частоти процесора, обсягу оперативної пам’яті, розміру постійної пам’яті, діагоналі екрану, кількості мега пікселів основної камери, ємності акумулятору. Також метою даного курсового проекту є набуття навичок з масштабування високонавантажених системи, роботою з Big Data, науковими бібліотеками мови програмування Python3 а також набуття навичок регресійного аналізу.

Аналіз інструментарію для виконання курсового проекту

Аналіз СУБД

Під час виконання курсового проекту виникла потреба зберігати велику кількість даних. Найкращий варіант для зберігання великої кількості даних-використання СУБД.

В якості СУБД були розглянуті варіанти: PostgreSQL, MongoDB. З порівняльною характеристикою цих СУБД можна ознайомитися в таблиці 1.

таблиця 1. Порівняльна характеристика СУБД

Критерій порівняння	Назва СУБД	
	MongoDB	PostgreSQL
Має відкритий вихідний код	так	так
Схема даних	динамічна	статична і динамічна
Підтримка ієрархічних даних	так	так (з 2012)
Реляційні дані	ні	так
Транзакції	ні	так
Атомарністі операцій	всередині документа	по всій БД
Мова запитів	JSON/JavaScript	SQL
Найлегший спосіб	горизонтальний	вертикальний

масштабування		
Підтримка шардингів	так	так (важка конфігурація)
Приклад використання	Великі дані (мільярди записів) з великою кількістю паралельних оновлень, де цілісність і узгодженість даних не потрібно.	Транзакційні і операційні програми, вигода яких в нормалізованому формі, об'єднаннях, обмеження даних і підтримки транзакцій.
Наявність бібліотек для мови програмування Python 3	так	так
Підтримка реплікації	так, автоматичне переобрання головного процесу	За принципом master-slave
Засіб збреження та відновлення даних	mongodump	pg_dump
Форма збереження даних	документи JSON	таблиця

За результатами порівняння цих СУБД було прийнято рішення зупинитися на NoSQL рішеннях. Оскільки вони чудово поєднують в собі переваги неструктурованих баз даних та простоту використання

горизонтального масштабування. Крім цього, класичним прикладом використання NoSQL СУБД є системи збору та аналізу даних, до яких можна застосувати індексування за первинними та вторинними ключами.

NoSQL база даних є об’єктно орієнтованою та дозволяє зберігати великі масиви неструктурованих даних. На відміну від SQL баз даних ми можемо зберігати дані у “сирому” об’єктному вигляді, який використовується програмою та є більш близьким за структурою до моделі даних, яку буде використовувати ПЗ написане з використанням мови програмування Python. Це пришвидшить збір, збереження та отримання даних програмним забезпеченням. Оскільки MongoDB є представником NoSQL баз даних, вона не потребує жорсткої схеми даних, що дозволяє пришвидшити процес розробки та зробити його більш гнучким. Окрім цього дана СУБД підтримує горизонтальне масштабування за допомогою шардингу з метою зменшення навантаження на кожен окремий вузол шляхом розподілення навантаження між ними. Також важливим є можливість використовувати готові рішення щодо засобів резервування/відновлення даних прямо «з коробки».

Обґрунтування вибору мови програмування

Мовою програмування для ПЗ було обрано Python 3.8. Ключовою особливістю Python є широкий інструментарій засобів розробки систем збору та аналізу даних. Фактично ця мова є стандартом у світі математичних розрахунків та обробки даних у реальному часі. Тож під час виконання курсового проекту було відносно легко знайти необхідну документацію та приклади роботи із цією мовою.

Обґрунтування вибору бібліотек і фреймворків

Використані бібліотеки та фреймворки:

- ***numpy*** - математична бібліотека мови Python, що додає підтримку великих багатовимірних масивів і матриць, разом з великою бібліотекою високорівневих математичних функцій для операцій з цими масивами. Використовується для роботи тільки з унікальними назвами та має широкий інструментарій у роботі з масивами (в даній роботі використовувався reshape)
- ***matplotlib*** - це бібліотека Python 2D, яка представляє числові дані у різноманітних форматах та інтерактивних середовищах на різних платформах (наприклад, зображення результатів аналізу у формі графіків).
- ***scikit-learn*** - бібліотека машинного навчання на мові програмування Python з відкритим вихідним кодом. Містить реалізації практично всіх можливих перетворень, і нерідко її однією вистачає для повної реалізації моделі. У бібліотеці також є основні алгоритми машинного навчання: *лінійної регресії* і її модифікацій Лассо, гребньовій регресії, опорних векторів вирішальних дерев і лісів та інше. (використовується для лінійної регресії)
- ***scrappy*** - це швидкий веб-фреймворк з відкритим вихідним кодом, написаний на Python, який використовується для отримання даних з веб-сторінки за допомогою селекторів на основі XPath (база даних заповнювалася за допомогою цього фреймворку).
- ***scipy*** - є відкритим вихідним кодом для Python, поширюваним в рамках ліцензованої бібліотеки BSD для виконання математичних, наукових та інженерних обчислень. Бібліотека SciPy створена для роботи з масивами NumPy і надає безліч зручних і ефективних чисельних методів, таких як процедури чисельної інтеграції та оптимізації.
- ***pymongo*** - є дистрибутивом Python, що містить інструменти для роботи з MongoDB, і є рекомендованим способом роботи з MongoDB від Python

(містить засоби реплікації, масштабування та резервування/відновлення даних прямо «з коробки»).

Структура бази даних

База даних складається з однієї колекції, в якій зберігаються характеристики смартфонів та їх ціна. Загальна структура документа в базі даних приведена у таблиці 2.

таблиця 2. Опис властивостей документа у базі даних

Назва властивості	Тип	Опис
_id	ObjectId	Ідентифікатор запису
name	String	Назва смартфону
memory	Double	Вбудована пам’ять смартфону
ram	Double	Оперативна пам’ять смартфону
diagonal	Double	Діагональ екрану смартфону
battery	Double	Ємність батареї смартфону
cpu_frequency	Double	Тактова частота процесора смартфону
camera	Double	Основна камера смартфону
price	Double	Ціна смартфону

Опис програмного забезпечення

Структура даного програмного забезпечення складається з 7 файлів, кожен з яких відповідає за важливу функцію даної системи.

- **data_science.py –** містить декілька алгоритмів побудови датасету сутностей даної бази даних, відповідає за правильну роботі із сформованим датасетом.
- **prediction.py –** містить декілька алгоритмів аналізу сутностей даної бази даних, аналізує дані, готові датафрейми та записує у відповідні файли результати аналізу даних.
- **database.py –** відповідає за доступ до документів колекції БД.
- **controller.py –** відповідає за користувацький інтерфейс та його безпомилкову роботу.
- **pipelines.py –** виконує «транспортування» даних до колекції БД;
- **FoxtrotSpider.py –** засіб «витягування» необхідних даних до БД;
- **MoyoSpider.py –** аналогічно попередньому файлу, але призначений для іншого ресурсу;

Опис модулів програмного забезпечення

У програмі можна виділити декілька основних логічних модулів, що мають певну самостійність і обмінюються один з одним даними. У програмі є 4 основні модулі: модуль генерації даних, модуль валідації, модулі аналітики та роботи з графіками та модуль бази даних. Деякі з них містять вкладені модулі. Взаємодія між модулями відбувається за допомогою виклику методів.

- **Модуль бази даних** містить функції, що виконують запис у базу даних з API.
- **Модуль валідації –** виконує валідацію бази даних на повторення записів що некорректно прибувають з API
- **Модулі аналітики та роботи з графіками** взаємодіє із згенерованими даними аналізуючи, впорядковуючи та візуалізуючи дані.
- **Модуль генерації даних** використовує дані з бази для того щоб згенерувати датафрейми з даними для аналізу корреляції даних.

Опис основних алгоритмів роботи

Алгоритм побудови графіків було реалізовано за допомогою потужного математичного пакету matplotlib. Найчастіше це перший пакет, пов'язаний з візуалізацією у Python. Спочатку задаємо назву графіку, потім встановлюємо панель управління, задаємо за які атрибути відповідають X та Y, підписуємо виведені дані та виводимо графік. Всю внутрішню реалізацію містять у собі функції математичного пакету matplotlib.

Алгоритм лінійної регресії було реалізовано за допомогою потужного наукового пакету scipy. Це інструмент-модель, що дозволяє візуалізувати лінійну залежність однієї змінної від набору інших змінних.

Аналіз функціонування засобів масштабування

Масштабування - можливість збільшити продуктивність проекту за мінімальний час шляхом додавання ресурсів. Зазвичай масштабування на увазі не переписування коду, а або додавання серверів, або нарощування ресурсів існуючого. З цього типу виділяють вертикальну та горизонтальну масштабування. Вертикальне - це коли додають більше оперативної пам'яті, дисків і т.д. на вже існуючий сервер, а горизонтальне - це коли ставлять більше серверів в дата-центри, і сервери там взаємодіють.

В базі даних MongoDB, як і в будь-якій NoSQL базі, не виникає проблем з горизонтальним масштабуванням, що і необхідно було у даному курсовому проекті для створення засобів резервування та відновлення даних, призначених для оперативного пакетного збереження фрагментів усієї бази даних з можливістю її відновлення з урахуванням необхідності підключення додаткового комп'ютера як елемента горизонтального масштабування (тобто додаткового сервера).

Реплікація - це процес синхронізації даних на декількох серверах. Даний механізм зменшує витрати ресурсів і збільшує доступність даних, копії яких зберігаються на різних серверах. Реплікація захищає базу даних від втрати єдиної сервера і дозволяє зберегти дані в разі технічної несправності на одному з серверів. У MongoDB реплікація досягається шляхом використання набору копій (replica set). Це група примірників mongod, який зберігають однакові набори даних. У копії один вузол - це ключовий вузол, який отримує всі операції запису. Всі інші вузли - вторинні, приймають операції з першого, таким чином, зберігаючи такі ж записи, як і первинний вузол. Набір копій може мати тільки один первинний вузол.

Реплікація MongoDB з Replica set

Суть ReplicaSet полягає в тому, що різні репліки в собі зберігають однакові набори даних. Один сервер повинен виступати в якості основного сервера (на нього будуть надходити всі дані - він же ведучий, він же PRIMARY), а всі інші - є вторинними (вони зберігають копії даних з PRIMARY - вони ж SECONDARYs).

Для правильно роботи ReplicaSet необхідно 3 запущених сервера з MongoDB:

- Одна нода виступатиме як арбітр і не братиме на себе ніякі дані. Її робота, полягає в тому, щоб вибрати того, хто буде сервером PRIMARY
- Один сервер виступає в якості PRIMARY сервера
- Один сервер виступає в якості SECONDARY сервера

Інструкція з репліціювання знаходиться у Додатку Б.

Шардінг - це процес зберігання документів на декількох серверах і це спосіб, яким MongoDB справляється з великими даними. З ростом кількості даних, один сервер не може зберігати всі даних, ні записувати їх, ні давати до них доступ. Шардінг вирішує проблему шляхом горизонтального масштабування. Завдяки даному механізму ми можемо підключати додаткові сервери для зберігання, записи і читання даних.

MongoDB sharding:

Шардована (shard - уламок) схема - схема, в якій дані не просто записуються в mongo чанки, які потім діляться навпіл, а потрапляють в чанки за певним діапазоном заданого поля - shard key. Спочатку створюється один чанк і діапазон значень, які приймає shard, він лежить в межах (-∞, + ∞). Коли розмір цього чанка досягає chunksize, mongos оцінює значення всіх shard keys всередині чанка і ділить чанк таким чином, щоб дані були розділені приблизно порівну.

Інструкція з шардування знаходиться у Додатку Б.

Опис результатів аналізу предметної галузі

В результаті виконання курсового проекту було проаналізовано дані про смартфони, що продаються в інтернет-магазинах. Було отримано наступні дані:

1. Згідно із рисунком А1

   Знайдено лінійну залежність ціни смартфонів від *вбудованої пам'яті смартфону*.
2. Згідно із рисунком А2

   Знайдено лінійну залежність ціни смартфонів від *оперативної пам'яті смартфону*.
3. Згідно із рисунком А3

   Передбачено ціну смартфонів, які містять такі характеристики, як 8Гб оперативної пам'яті, 128Гб вбудованої пам'яті, тактову частоту процесору 2,2 ГГц та ємність батареї 5000мА*г.
4. Згідно із рисунком А4

   Знайдено коефіцієнти «профітності» смартфонів в залежності від заданих характерстик і ціни та формування топу найкращих/найгірших смартфонів за критерієм «ціна/характеристики».

Висновки

В процесі виконання даного курсового проекту було отримано практичні навички обробки великих масивів даних за допомогою мови програмування Python 3 та СУБД MongoDB.

Була успішно опрацьована відповідна технічна література для успішного написання курсового проекту (перелік літератури наведено нижче). В ході розробки даного програмного забезпечення були додані засоби валідації та фільтрації даних з достатньо великою швидкістю. Були додані засоби аналізу даних та графічне представлення результатів за допомогою існуючих математичних пакетів мови Python. Програмне забезпечення було успішно протестовано на поки що на одному комп’ютері, швидкість роботи бази даних достатньо велика. Також було реалізовано реплікації бази даних, бекап бази даних і шардування.

В ході виконання даного курсового проекту було досягнуто поставленої мети: було набуто практичних навичок розробки сучасного програмного забезпечення, що взаємодіє з базами даних, а також були здобуті навички оформлення відповідного текстового, програмного та ілюстративного матеріалу у формі проектної документації.

Література

1. MongoDB от теории к практике. Руководство по установке кластера mongoDB [Електронний ресурс]. – Режим доступу до ресурсу: https://m.habr.com/ru/post/217393/;
2. Почему Python так хорош в научных вычислениях [Електронний ресурс]. – Режим доступу до ресурсу: https://habr.com/post/349482/;
3. Scrapy: собираем данные и сохраняем в базу данных [Електронний ресурс]. – Режим доступу до ресурсу: https://habr.com/ru/post/308660/
4. Python [Електронний ресурс]. – Режим доступу до ресурсу: https://devguide.python.org;
5. Scikit-learn [Електронний ресурс]. – Режим доступу до ресурсу: https://scikit-learn.org/stable/getting_started.html;
6. Matplotlib [Електронний ресурс]. – Режим доступу до ресурсу: https://matplotlib.org/stable/contents.html;
7. NumPy [Електронний ресурс]. – Режим доступу до ресурсу: https://numpy.org/doc/stable/;

Додаток А

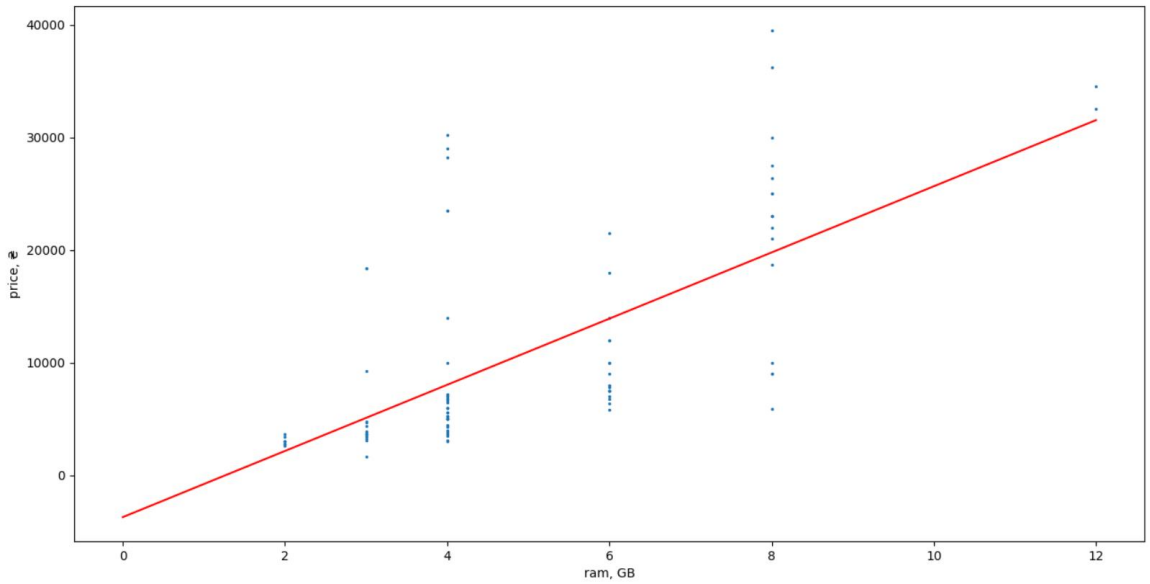

Рис. А1 Лінійна регресія залежності ціни смартфонів від вбудованої пам'яті смартфону

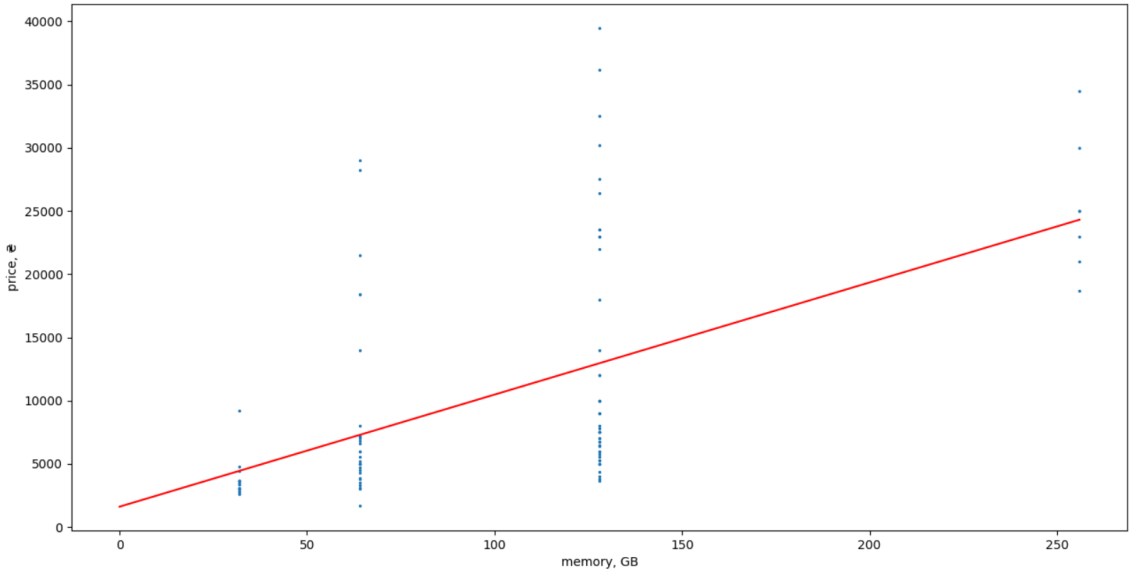

Рис. А2 Лінійна регресія залежності ціни смартфонів від оперативної пам'яті смартфону

```
>>> multivariate ram memory cpu_frequency battery
(GB) ram = 8
(GB) memory = 128
(GHz) cpu_frequency = 2.2
(mAh) battery = 5000
price = 14,415 ₴
```

*Рис. А3 Передбачення ціни смартфонів, які містять такі характеристики, як 8Гб оперативної пам'яті, 128Гб вбудованої пам'яті, тактову частоту процесору 2,2 ГГц та ємність батареї 5000мА*г.*

```
>>> profit
                                    name    price  predicted price    profit
27              MEIZU M10 3/32GB Dual Sim   3499.0            10777  3.080023
33             NOKIA 3.4 3/64 Gb Dual Sim   1699.0             4799  2.824603
44                      OPPO A91 8/128GB   5949.0            16597  2.789881
104                Xiaomi Redmi 9C 2/32GB   3099.0             8371  2.701194
39              OPPO A31 4/64 Gb Dual Sim   4499.0            10873  2.416759
..                                    ...      ...              ...       ...
56           SAMSUNG Galaxy A31 4/64 Duos   5999.0             2294  0.382397
57    SAMSUNG Galaxy A32 4/64 Gb Dual Sim   6599.0             2484  0.376421
52                       REALME C3 2/32Gb   3399.0             1219  0.358635
17                CAT S61 4/64 Gb Dual Sim  28999.0             9452  0.325942
101  XIAOMI Redmi Note 9 3/64 Gb Dual Sim   4699.0               16  0.003405
```

Рис. А4 Топ найкращих/найгірших смартфонів за критерієм «ціна/характеристики»

Додаток Б

Інструкція з репліціювання:

```
# Запускаємо сервер в якості PRIMARY (master)
“C:\Program Files\MongoDB\Server\4.4\bin\mongod.exe” --dbpath=”c:\data\db1” --port 27001 --replSet r0

# Запускаємо сервер в якості SECONDARY (slave):
“C:\Program Files\MongoDB\Server\4.4\bin\mongod.exe” --dbpath=”c:\data\db2” --port 27002 --replSet r0

# Запускаємо сервер в якості арбітра (який не зберігає даних):
“C:\Program Files\MongoDB\Server\4.4\bin\mongod.exe” --dbpath=”c:\data\db3” --port 27003 --replSet r0

# Налаштування PRIMARY сервера:
“C:\Program Files\MongoDB\Server\4.4\bin\mongo.exe” --host 127.0.0.1 --port 27001
rs.initiate({"_id" : "My_Replica_Set", members : [ {"_id" : 0, priority : 3, host : "127.0.0.1:27001"}, {"_id" : 1, host : "127.0.0.1:27002"}, {"_id" : 2, host : "127.0.0.1:27003", arbiterOnly : true} ] });
```

В результаті налаштована MongoDB буде доступна за адресою mongodb://127.0.0.1:27003:

```
> rs.initiate({"_id" : "r0", members : [ {"_id" : 0, priority : 3, host : "127.0.0.1:27001"}, {"_id" : 1, host : "127.0.0.1:27002"}, {"_id" : 2, host : "127.0.0.1:27003", arbiterOnly : true} ] });
{ "ok" : 1 }
r0:SECONDARY> rs.status()
{
        "set" : "r0",
        "date" : ISODate("2021-05-17T14:47:44.066Z"),
        "myState" : 1,
        "term" : NumberLong(1),
        "syncSourceHost" : "",
        "syncSourceId" : -1,
        "heartbeatIntervalMillis" : NumberLong(2000),
        "majorityVoteCount" : 2,
        "writeMajorityCount" : 2,
        "votingMembersCount" : 3,
        "writableVotingMembersCount" : 2,
        "optimes" : {
                "lastCommittedOpTime" : {
                        "ts" : Timestamp(1621262862, 1),
                        "t" : NumberLong(1)
```

```
                },
                "lastCommittedWallTime" : ISODate("2021-05-17T14:47:42.447Z"),
                "readConcernMajorityOpTime" : {
                        "ts" : Timestamp(1621262862, 1),
                        "t" : NumberLong(1)
                },
                "readConcernMajorityWallTime" : ISODate("2021-05-
17T14:47:42.447Z"),
                "appliedOpTime" : {
                        "ts" : Timestamp(1621262862, 1),
                        "t" : NumberLong(1)
                },
                "durableOpTime" : {
                        "ts" : Timestamp(1621262862, 1),
                        "t" : NumberLong(1)
                },
                "lastAppliedWallTime" : ISODate("2021-05-17T14:47:42.447Z"),
                "lastDurableWallTime" : ISODate("2021-05-17T14:47:42.447Z")
        },
        "lastStableRecoveryTimestamp" : Timestamp(1621262860, 3),
        "electionCandidateMetrics" : {
                "lastElectionReason" : "electionTimeout",
                "lastElectionDate" : ISODate("2021-05-17T14:47:40.209Z"),
                "electionTerm" : NumberLong(1),
                "lastCommittedOpTimeAtElection" : {
                        "ts" : Timestamp(0, 0),
                        "t" : NumberLong(-1)
                },
                "lastSeenOpTimeAtElection" : {
                        "ts" : Timestamp(1621262849, 1),
                        "t" : NumberLong(-1)
                },
                "numVotesNeeded" : 2,
                "priorityAtElection" : 3,
                "electionTimeoutMillis" : NumberLong(10000),
                "numCatchUpOps" : NumberLong(0),
                "newTermStartDate" : ISODate("2021-05-17T14:47:40.535Z"),
                "wMajorityWriteAvailabilityDate" : ISODate("2021-05-
17T14:47:41.651Z")
        },
        "members" : [
                {
                        "_id" : 0,
                        "name" : "127.0.0.1:27001",
                        "health" : 1,
                        "state" : 1,
                        "stateStr" : "PRIMARY",
                        "uptime" : 81,
                        "optime" : {
                                "ts" : Timestamp(1621262862, 1),
                                "t" : NumberLong(1)
                        },
                        "optimeDate" : ISODate("2021-05-17T14:47:42Z"),
                        "syncSourceHost" : "",
                        "syncSourceId" : -1,
                        "infoMessage" : "Could not find member to sync from",
                        "electionTime" : Timestamp(1621262860, 1),
                        "electionDate" : ISODate("2021-05-17T14:47:40Z"),
                        "configVersion" : 1,
                        "configTerm" : 1,
                        "self" : true,
```

```
                        "lastHeartbeatMessage" : ""
                },
                {
                        "_id" : 1,
                        "name" : "127.0.0.1:27002",
                        "health" : 1,
                        "state" : 2,
                        "stateStr" : "SECONDARY",
                        "uptime" : 14,
                        "optime" : {
                                "ts" : Timestamp(1621262860, 5),
                                "t" : NumberLong(1)
                        },
                        "optimeDurable" : {
                                "ts" : Timestamp(1621262860, 5),
                                "t" : NumberLong(1)
                        },
                        "optimeDate" : ISODate("2021-05-17T14:47:40Z"),
                        "optimeDurableDate" : ISODate("2021-05-17T14:47:40Z"),
                        "lastHeartbeat" : ISODate("2021-05-17T14:47:42.366Z"),
                        "lastHeartbeatRecv" : ISODate("2021-05-17T14:47:43.442Z"),
                        "pingMs" : NumberLong(0),
                        "lastHeartbeatMessage" : "",
                        "syncSourceHost" : "127.0.0.1:27001",
                        "syncSourceId" : 0,
                        "infoMessage" : "",
                        "configVersion" : 1,
                        "configTerm" : 1
                },
                {
                        "_id" : 2,
                        "name" : "127.0.0.1:27003",
                        "health" : 1,
                        "state" : 7,
                        "stateStr" : "ARBITER",
                        "uptime" : 14,
                        "lastHeartbeat" : ISODate("2021-05-17T14:47:42.366Z"),
                        "lastHeartbeatRecv" : ISODate("2021-05-17T14:47:42.438Z"),
                        "pingMs" : NumberLong(0),
                        "lastHeartbeatMessage" : "",
                        "syncSourceHost" : "",
                        "syncSourceId" : -1,
                        "infoMessage" : "",
                        "configVersion" : 1,
                        "configTerm" : 1
                }
        ],
        "ok" : 1,
        "$clusterTime" : {
                "clusterTime" : Timestamp(1621262862, 1),
                "signature" : {
                        "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),
                        "keyId" : NumberLong(0)
                }
        },
        "operationTime" : Timestamp(1621262862, 1)
}
```

Інструкція з шардування:

```
# Створимо  репліка сет №1
“C:\Program Files\MongoDB\Server\4.4\bin\mongod.exe” --port 27000--replSet r0 --host localhost
“C:\Program  Files\MongoDB\Server\4.4\bin\mongod.exe” --dbpath=”c:\data\db1” --port 27001 --replSet r0 --bind_ip localhost
“C:\Program  Files\MongoDB\Server\4.4\bin\mongod.exe” --dbpath=”c:\data\db2” --port 27002 --replSet r0 --bind_ip localhost
“C:\Program Files\MongoDB\Server\4.4\bin\mongo.exe” --bind_ip localhost --port 27000
rs.initiate()
rs.conf()
rs.add("localhost:27001")
rs.add("localhost:27002")
rs.status()
rs.isMaster()

# Створимо  репліка сет №2
“C:\Program  Files\MongoDB\Server\4.4\bin\mongod.exe” --dbpath=”c:\data\db3” --port 27003 --replSet r1 --bind_ip localhost
“C:\Program  Files\MongoDB\Server\4.4\bin\mongod.exe” --dbpath=”c:\data\db4” --port 27004 --replSet r1 --bind_ip localhost
“C:\Program  Files\MongoDB\Server\4.4\bin\mongod.exe” --dbpath=”c:\data\db5” --port 27005 --replSet r1 --bind_ip localhost
“C:\Program Files\MongoDB\Server\4.4\bin\mongo.exe” --bind_ip localhost --port 27003
rs.initiate()
rs.conf()
rs.add("localhost:27004")
rs.add("localhost:27005")

# Піднімаємо  конфіг сет
“C:\Program Files\MongoDB\Server\4.4\bin\mongod.exe” --configsvr --replSet configReplSet --port 28000 --dbpath="c:\data\db7" --bind_ip localhost
“C:\Program Files\MongoDB\Server\4.4\bin\mongod.exe” --configsvr --replSet configReplSet --port 28001 --dbpath="c:\data\db8" --bind_ip localhost
“C:\Program Files\MongoDB\Server\4.4\bin\mongod.exe” --configsvr --replSet configReplSet --port 28002 --dbpath="c:\data\db9" --bind_ip localhost
```

```
# Піднімаємо mongos
“C:\Program Files\MongoDB\Server\4.4\bin\mongos.exe” --configdb
configReplSet/localhost:28000,localhost:28001,localhost:28002 --port 29000
“C:\Program Files\MongoDB\Server\4.4\bin\mongo.exe” --port 29000 --bind_ip
localhost
rs.initiate({ _id: "configReplSet",  configsvr: true,
               members:[
                          { _id: 0, host: "localhost:28000" },
                          { _id: 1, host: "localhost:28001" },
                          { _id: 2, host: "localhost:28002" },
               ]})
sh.addShard("r0/localhost:27000,localhost:27001,localhost:27002")
sh.addShard("r1/localhost:27003,localhost:27004,localhost:27005")
use admin
sh.enableSharding("course_work")
# вмикаємо шардування за полем price
db.runCommand({shardCollection: "course_work.smartphones", key: {price: 1}})
# перевіряємо результат
db.printShardingStatus()
```

В результаті налаштована MongoDB буде доступна за адресою mongodb://localhost:29000:

```
mongos> db.printShardingStatus(undefined,true)
--- Sharding Status ---
sharding version: {
         "_id" : 1,
         "currentVersion" : 4
}
  shards:
        { "_id" : "shard0000", "host" : "localhost:27000" }
        { "_id" : "shard0001", "host" : "localhost:27003" }
databases:
        { "_id" : "admin", "partitioned" : false, "primary" : "config" }
        { "_id" : "smartphones", "partitioned" : true, "primary" : "shard0000" }
                  smartphones.cases
                           shard key: { "price" : 1 }
                           chunks:
                                    shard0000           4
                                    shard0001           3
```

Засоби резервування/відновлення даних:

```
# резервування
mongodump --dbpath=”c:\data\db\” –out=”c:\data\backups\db\”
# відновлення
mongorestore --dbpath=”c:\data\db\” ”c:\data\backups\db\”
```