факультет информатики, математики и компьютерных наук

Компьютерные науки и технологии

Управление данными

Лабораторная работа №7 **MongoDB**

Студенты- участники : Гайфиев Иван
Булычев Андрей
Староверов Роман
Деменьтев Леонид
Агафонов Алексей
Николаев Никита

Введение в MongoDB

MongoDB — это документоориентированная NoSQL-система управления базами данных, разработанная для работы с большими объемами неструктурированных и полуструктурированных данных

История создания:

MongoDB была создана в 2007 году компанией MongoDB Inc и разрабатывалась как облачная платформа , но позже стала самостоятельным продуктом

Основная цель создания:

Необходимость в альтернативе традиционным реляционным базам данных, которые не всегда справлялись с растущими требованиями к гибкости и масштабируемости.

Основные особенности MongoDB

- MongoDB относится к документоориентированным NoSQL-системам.
- Гибкая схема данных (schema-less): MongoDB не требует строгой схемы данных, что позволяет легко изменять структуру документов.
- Горизонтальная масштабируемость (шардирование): Данные могут быть распределены по нескольким серверам для повышения производительности.
- Репликация: MongoDB поддерживает репликацию данных для обеспечения отказоустойчивости и высокой доступности.
- Индексация: Поддержка индексов для ускорения поиска данных.

Преимущества MongoDB в сравнении с SQL

•**Гибкость структуры данных**: В MongoDB можно хранить данные с разной структурой в одной коллекции, что невозможно в SQL.

•**Высокая производительность**: MongoDB оптимизирована для работы с большими объемами данных и сложными запросами.

•**Масштабируемость**: Горизонтальное масштабирование (шардирование) позволяет распределять данные по нескольким серверам.

Недостатки MongoDB в сравнении с SQL

•**Отсутствие транзакций на уровне нескольких документов**: В ранних версиях MongoDB не поддерживались транзакции, но в современных версиях эта проблема решена.

•**Высокие требования к оперативной памяти**: MongoDB активно использует оперативную память для хранения индексов и данных.

Сравнение с другими NoSQL-решениями:

•**Cassandra**: Подходит для работы с большими объемами данных и высокой нагрузкой, но менее гибкая в плане структуры данных.

•**Redis**: Оптимизирован для работы с ключ-значение и кэшированием, но не подходит для сложных запросов.

•**Elasticsearch**: Используется для полнотекстового поиска, но не подходит для хранения структурированных данных.

Основные функциональные возможности MongoDB

CRUD-операции:

Create (создание): Добавление новых документов в коллекцию.

В нашем коде это реализовано в функции add_record():

python

```
def add_record():
    collection = db["job"]
    number = int(input("Введите номер договора: "))
    date = input("Введите дату (месяц): ")
    id_employee = int(input("Введите ID сотрудника: "))
    id_place_of_work = int(input("Введите ID места работы: "))
    id_job_title = int(input("Введите ID должности: "))
    num_hours = int(input("Введите количество часов: "))
    salary_rub = float(input("Введите сумму оплаты (в рублях): "))

    record = {
        "number": number,
        "date": date,
        "id_employee": id_employee,
        "id_place_of_work": id_place_of_work,
        "id_job_title": id_job_title,
        "num_hours": num_hours,
        "salary_rub": salary_rub
    }
    collection.insert_one(record)
    print("Запись добавлена.")
```

Основные особенности MongoDB

Read (чтение): Поиск и извлечение данных из коллекции.

В нашем коде это реализовано в функции show_records():

```
def show_records():
    collection = db["job"]
    records = collection.find()
    for record in records:
        print(record)
```

Основные особенности MongoDB

Update (обновление): Изменение существующих документов.

В нашем коде это реализовано в функции update_record():

```
def update_record():
    collection = db["job"]
    number_of_contract = int(input("Введите номер договора(поле 'number') для обновления: "))
    new_salary = float(input("Введите новую сумму оплаты: "))

    result = collection.update_one({"number": number_of_contract}, {"$set": {"salary_rub": new_salary}})
    if result.matched_count:
        print("Запись успешно обновлена.")
    else:
        print("Запись не найдена.")
```

Основные особенности MongoDB

Delete (удаление): Удаление документов из коллекции.

В нашем коде это реализовано в функции delete_record():

```
def delete_record():
    collection = db["job"]
    number_of_contract = int(input("Введите номер договора(поле 'number') для удаления: "))

    result = collection.delete_one({"number": number_of_contract})
    if result.deleted_count:
        print("Запись успешно удалена.")
    else:
        print("Запись не найдена.")
```

Основные особенности MongoDB

Очистка коллекции:
В нашем коде это реализовано в функции clear_collection():

```
def clear_collection():
    collection = db["job"]
    confirm = input("Вы уверены, что хотите удалить ВСЕ записи? (y/n): ")
    if confirm.lower() == "y":
        collection.delete_many({})
        print("Все записи удалены!")
    else:
        print("Очистка отменена.")
```

Создание коллекции:
В нашем коде это реализовано в функции create_collection():

```
def create_collection():
    if "new_collection" not in db.list_collection_names():
        db.create_collection("new_collection")
        print("Коллекция создана.")
    else:
        print("Коллекция уже существует.")
```

Возможность интеграции со сторонними приложениями

В нашем коде используется библиотека **pymongo** для работы с **MongoDB** из **Python :**

```
from pymongo import MongoClient
client =
MongoClient("mongodb://localhost:27017/")
db = client["DM_lab_7"]
```

*пример интеграции **MongoDB** с **Python**-приложением

Что требуется для реализации доступа?

Для доступа к **MongoDB** через **Python** необходимо:

Установить библиотеку **pymongo:**

```
pip install pymongo
```

Затем Подключиться к **MongoDB** с использованием строки подключения:

```
client = MongoClient("mongodb://localhost:27017/")
db = client["DM_lab_7"]
```

и выполнять запросы к базе данных с помощью **API pymongo.**

Заключение

- MongoDB — это мощный инструмент для работы с большими объемами данных, который предлагает гибкость и высокую производительность.
- MongoDB легко интегрируется с Python через библиотеку pymongo, что делает её удобным выбором для разработки современных приложений.
- В сравнении с SQL, MongoDB предлагает более гибкую структуру данных и высокую производительность при работе с большими объемами информации.


ВЫСШАЯ • ШКОЛА • ЭКОНОМИКИ •