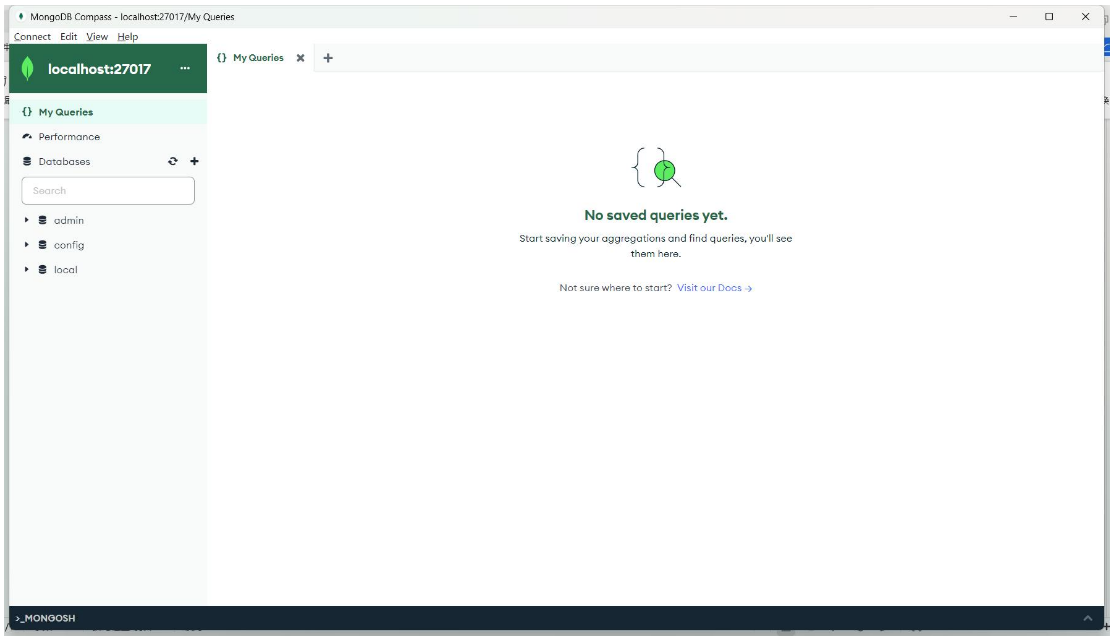
“大数据工程”课程实验报告

题目：网络爬虫的综合编程实验	学号：21377061 姓名：范春	日期：2024 年 4 月 19 日

实验环境：

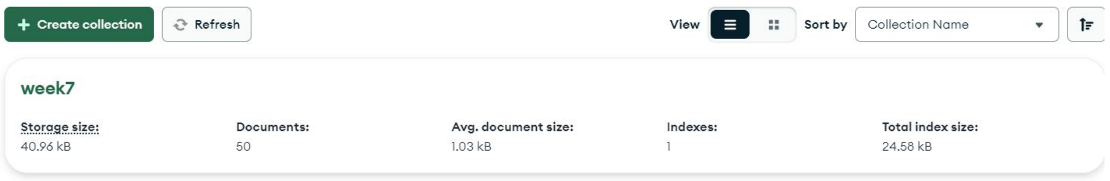
本次实验在 Windows 上进行，MongoDB 配制如下：


MongoDB Compass - localhost:27017/My Queries
Connect Edit View Help
localhost:27017
My Queries
Performance
Databases
Search
admin
config
local
No saved queries yet.
Start saving your aggregations and find queries, you'll see them here.
Not sure where to start? Visit our Docs →
>_MONGOSH

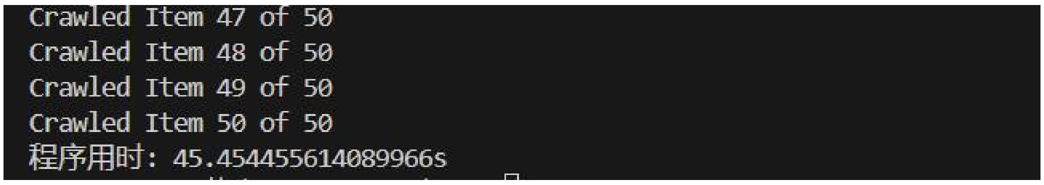
实验内容与完成情况：

任务 1：

本次实验爬取了 50 条数据，结果如下：



week7				
Storage size:	Documents:	Avg. document size:	Indexes:	Total index size:
40.96 kB	50	1.03 kB	1	24.58 kB



```
Crawled Item 47 of 50
Crawled Item 48 of 50
Crawled Item 49 of 50
Crawled Item 50 of 50
程序用时: 45.454455614089966s
```

完整代码如下：

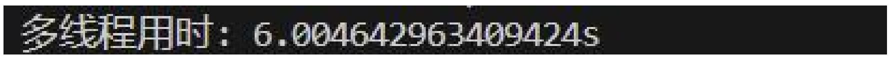
```python
import json
import time
from pymongo import MongoClient
from config import *
import requests


class Scraper:
    @staticmethod
    def get_homepage(category_id, page, page_size):
        url = "https://apipc-xiaotuxian-front.itheima.net/category/goods/temporary"
        headers = {
            "content-type": "application/json",
            "user-agent": "HomeworkCrawler/1.0",
        }
        data = {
            'categoryId': str(category_id),
            'page': str(page),
            'pageSize': str(page_size)
        }
        jsondata = json.dumps(data)
        resp = requests.post(url=url, data=jsondata, headers=headers)
        resp_json = json.loads(resp.text)
        return resp_json['result']
    @staticmethod
    def get_detailpage(item_id):
        url = f"https://apipc-xiaotuxian-front.itheima.net/goods?id={item_id}"
        headers = {
            "user-agent": "HomeworkCrawler/1.0",
        }
        resp = requests.get(url, headers=headers)
        resp_json = json.loads(resp.text)
        return resp_json['result']
    @staticmethod
    def get_goods_picture(url):
        pic_name = url.split("/")[-1]
        resp = requests.get(url)
        with open(f"picture/{pic_name}", 'wb') as f:
            f.write(resp.content)
        return pic_name
    @staticmethod
    def analyze(homepage_info: dict, detail_info: dict, pic: str):
        item_info = {
            'id': int(homepage_info['id']),
            'url': f"https://erabbit.itheima.net/#/product/{homepage_info['id']}",
            'name': homepage_info['name'],
            'desc': homepage_info['desc'],
            'price': float(homepage_info['price']),
            'pic': pic,
            'detail': str(detail_info['details']['properties'])
        }
        return item_info
    @staticmethod
```

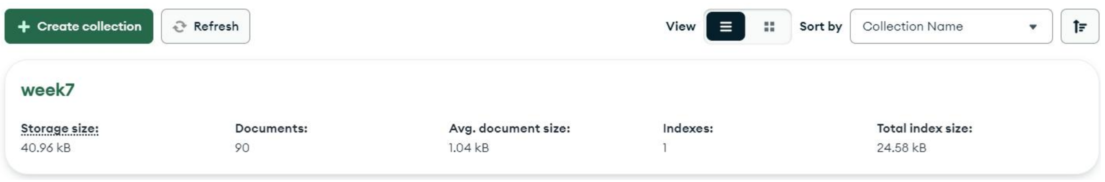
```python
    def crawl_and_store(category_id, num, collection):
        all_items = []
        item_detail = {}
        pics = {}
        for i in range(1, (num+19)//20 + 1):
            res = None
            while res is None:
                res = Scraper.get_homepage(category_id, i, 20)
            all_items.extend(res['items'])
            print(f"Crawled Page {i} of {(num+19)//20}")
        for itemid, item in enumerate(all_items[:num]):
            detail = None
            pic = None
            while detail is None:
                detail = Scraper.get_detailpage(item['id'])
            while pic is None:
                pic = Scraper.get_goods_picture(item['picture'])
            item_detail[item['id']] = detail
            pics[item['id']] = pic
            print(f"Crawled Item {itemid + 1} of {num}")
        for item in all_items[:num]:
            analyzed_item = Scraper.analyze(item, item_detail[item['id']], pics[item['id']])
            collection.insert_one(analyzed_item)
if __name__ == '__main__':
    print("开始爬取")
    start = time.time()
    category = "109243036"
    client = MongoClient('mongodb://localhost:27017')
    db = client['week7']
    collection = db['week7']
    print("数据库连接成功")
    Scraper.crawl_and_store(category, 50, collection)
    end = time.time()
    print(f"程序用时: {end-start}s")
```

任务 2:

(1) 多线程：爬取了两页数据，结果如下



```
多线程用时: 6.004642963409424s
```



week7

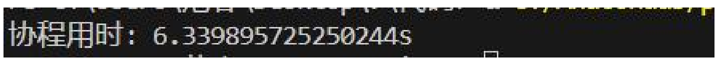
Storage size: 40.96 kB
Documents: 90
Avg. document size: 1.04 kB
Indexes: 1
Total index size: 24.58 kB

完整代码如下：

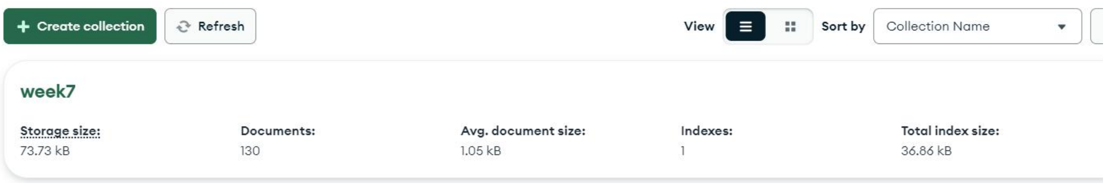
```python
import time
import config
from threading import Lock, Thread
from single_thread import Scraper
import queue
from pymongo import MongoClient

class MultithreadScraper:
    def __init__(self, collection, crawl_pages=2, category_id=109243036):
        self._homepage_queue = queue.Queue()
        self._detail_info_queue = queue.Queue()
        self._picture_queue = queue.Queue()
        self._base_info_lock = Lock()
        self._base_info = []
        self._pic_info_lock = Lock()
        self._pic_info = {}
        self._detail_info_lock = Lock()
        self._detail_info = {}
        self._collection = collection
        self._crawl_pages = crawl_pages
        self._crawl_category_id = category_id
    def _add_work_to_homepage_queue(self):
        for i in range(1, self._crawl_pages + 1):
            self._homepage_queue.put((self._crawl_category_id, i, 20))
    def work_homepage_queue(self):
        param = self._homepage_queue.get()
        result = Scraper.get_homepage(*param)
        time.sleep(0.5)
        for i in result['items']:
            self._picture_queue.put((i['id'], i['picture']))
            self._detail_info_queue.put((i['id'],))
            with self._base_info_lock:
                self._base_info.append(i)
    def work_picture_queue(self):
        param = self._picture_queue.get()
        result = Scraper.get_goods_picture(param[1])
        time.sleep(0.5)
        with self._pic_info_lock:
            self._pic_info[param[0]] = result
    def work_detail_queue(self):
        param = self._detail_info_queue.get()
        result = Scraper.get_detailpage(param[0])
        time.sleep(0.5)
        with self._detail_info_lock:
            self._detail_info[param[0]] = result
    def _write_to_db(self):
        for i in self._base_info:
            id = i['id']
            data = Scraper.analyze(
                homepage_info=i,
                detail_info=self._detail_info[id],
                pic=self._pic_info[id]
            )
            self._collection.insert_one(data)
    def exec(self):
        self._add_work_to_homepage_queue()
        homepage_threads = []
        for i in range(self._homepage_queue.qsize()):
            homepage_threads.append(Thread(target=self.work_homepage_queue))
        for i in homepage_threads:
            i.start()
        for i in homepage_threads:
            i.join()
        pic_detail_threads = []
        for i in range(self._picture_queue.qsize()):
            pic_detail_threads.append(Thread(target=self.work_picture_queue))
        for i in range(self._detail_info_queue.qsize()):
            pic_detail_threads.append(Thread(target=self.work_detail_queue))
        for i in pic_detail_threads:
            i.start()
        for i in pic_detail_threads:
            i.join()
        self._write_to_db()
if __name__ == '__main__':
    start = time.time()
    client = MongoClient('mongodb://localhost:27017')
    db = client['week7']
    collection = db['week7']
    multicrawler = MultithreadScraper(collection)
    multicrawler.exec()
    end = time.time()
    print(f"多线程用时: {end-start}s")
```

(2) 协程：仍然爬取了两页数据，结果如下



```
协程用时: 6.339895725250244s
```



week7

Storage size: 73.73 kB
Documents: 130
Avg. document size: 1.05 kB
Indexes: 1
Total index size: 36.86 kB

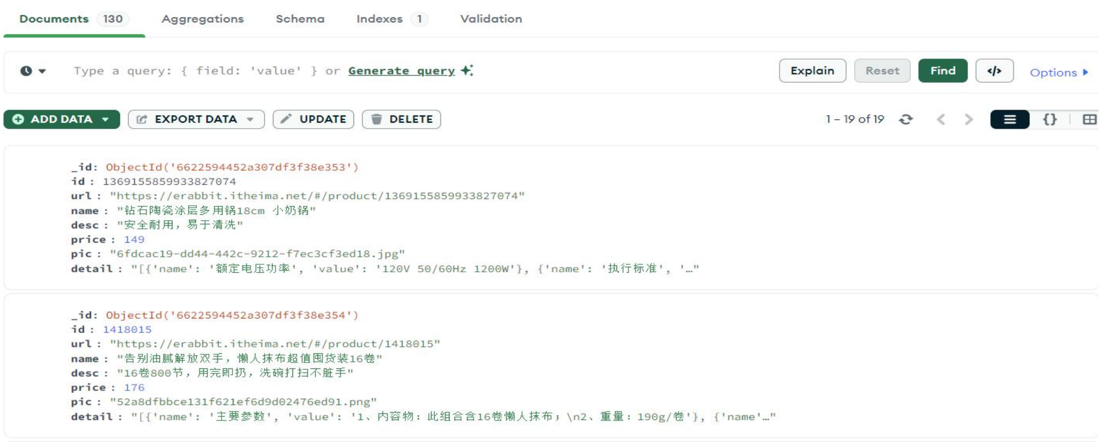
完整代码如下：

```python
import asyncio
import json
import time
import aiohttp
from config import *
from pymongo import MongoClient
from single_thread import Scraper

class CoroutineScraper:
    @staticmethod
    async def get_homepage(category_id, page, page_size):
        url = "https://apipc-xiaotuxian-front.itheima.net/category/goods/temporary"
        headers = {
            "content-type": "application/json",
            "user-agent": "HomeworkCrawler/1.0",
        }
        data = {
            'categoryId': str(category_id),
            'page': str(page),
            'pageSize': str(page_size)
        }
        jsondata = json.dumps(data)
        conn = aiohttp.TCPConnector(ssl=False)
        async with aiohttp.ClientSession(connector=conn) as session:
            async with await session.post(url=url, data=jsondata, headers=headers) as response:
                resp = await response.text()
        resp_json = json.loads(resp)
        return resp_json['result']
    @staticmethod
    async def get_detailpage(item_id):
        url = f"https://apipc-xiaotuxian-front.itheima.net/goods?id={item_id}"
        headers = {
            "user-agent": "HomeworkCrawler/1.0",
        }
        conn = aiohttp.TCPConnector(ssl=False)
        async with aiohttp.ClientSession(connector=conn) as session:
            async with await session.get(url=url, headers=headers) as response:
                resp = await response.text()
        resp_json = json.loads(resp)
        return item_id, resp_json['result']
    @staticmethod
    async def get_item_picture(item_id, url):
        pic_name = url.split("/")[-1]
        cache_path = f"cache/pic_{pic_name}.cache"
        conn = aiohttp.TCPConnector(ssl=False)
        async with aiohttp.ClientSession(connector=conn) as session:
            async with await session.get(url=url) as response:
                resp_content = await response.content.read()
        with open(f"picture/{pic_name}", 'wb') as f:
            f.write(resp_content)
        return item_id, pic_name
    @staticmethod
    def analyze_item(homepage_info: dict, detail_info: dict, pic: str):
        return Scraper.analyze(homepage_info, detail_info, pic)
    @staticmethod
    async def crawl_and_store(category_id, num_pages, connection):
        all_items = []
        item_detail = {}
        pics = {}
        homepage_tasks = []
        for i in range(1, num_pages + 1):
            homepage_tasks.append(
                asyncio.create_task(
                    CoroutineScraper.get_homepage(category_id, i, 20)
                )
            )
        await asyncio.gather(*homepage_tasks)
        for i in homepage_tasks:
            ires = i.result()
            all_items.extend(ires['items'])
        detail_tasks = []
        pic_tasks = []
        for item in all_items:
            item_id = item['id']
            pic_url = item['picture']
            detail_tasks.append(
                asyncio.create_task(
                    CoroutineScraper.get_detailpage(item_id)
                )
            )
            pic_tasks.append(
                asyncio.create_task(
                        CoroutineScraper.get_item_picture(item_id, pic_url)
                    )
                )
            await asyncio.gather(*detail_tasks)
            await asyncio.gather(*pic_tasks)
            for i in detail_tasks:
                ires = i.result()
                item_detail[ires[0]] = ires[1]
            for i in pic_tasks:
                ires = i.result()
                pics[ires[0]] = ires[1]
            for item in all_items:
                connection.insert_one(CoroutineScraper.analyze_item(item,
item_detail[item['id']], pics[item['id']]))
        @staticmethod
        def run(category_id, num_pages, connection):
            loop = asyncio.get_event_loop()
            coroutine = asyncio.ensure_future(
                CoroutineScraper.crawl_and_store(category_id, num_pages, connection)
            )
            loop.run_until_complete(asyncio.gather(coroutine))
    if __name__ == '__main__':
        start = time.time()
        category = "109243036"
        client = MongoClient('mongodb://localhost:27017')
        db = client['week7']
        connection = db['week7']
        CoroutineScraper.run(category, 2, connection)
        end = time.time()
        print(f"协程用时: {end-start}s")
```
由上述运行结果可知，多线程和协程的爬取速度远远高于单线程。

任务 3：

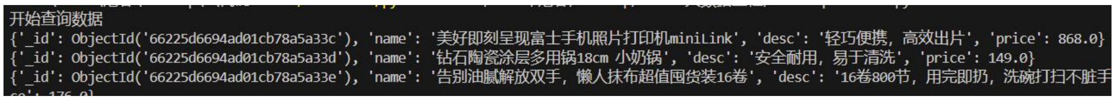
MongoDB 中数据详情如下所示：



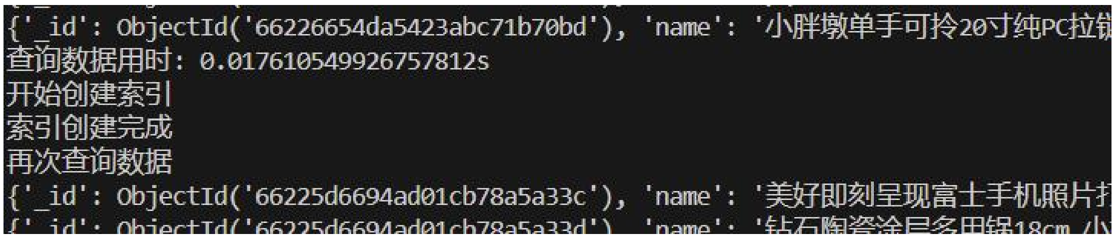
```
_id: ObjectId('6622594452a307df3f38e353')
id : 1369155859933827074
url : "https://erabbit.itheima.net/#/product/1369155859933827074"
name : "钻石陶瓷涂层多用锅18cm 小奶锅"
desc : "安全耐用，易于清洗"
price : 149
pic : "6fdcac19-dd44-442c-9212-f7ec3cf3ed18.jpg"
detail : "[{'name': '额定电压功率', 'value': '120V 50/60Hz 1200W'}, {'name': '执行标准', '…"

_id: ObjectId('6622594452a307df3f38e354')
id : 1418015
url : "https://erabbit.itheima.net/#/product/1418015"
name : "告别油腻解放双手，懒人抹布超值囤货装16卷"
desc : "16卷800节，用完即扔，洗碗打扫不脏手"
price : 176
pic : "52a8dfbbce131f621ef6d9d02476ed91.png"
detail : "[{'name': '主要参数', 'value': '1、内容物：此组合含16卷懒人抹布；\n2、重量：190g/卷'}, {'name'…"
```

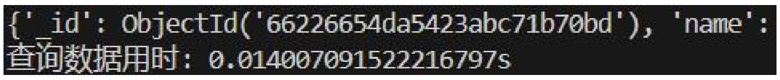
运行结果如下：



```
开始查询数据
{'_id': ObjectId('66225d6694ad01cb78a5a33c'), 'name': '美好即刻呈现富士手机照片打印机miniLink', 'desc': '轻巧便携，高效出片', 'price': 868.0}
{'_id': ObjectId('66225d6694ad01cb78a5a33d'), 'name': '钻石陶瓷涂层多用锅18cm 小奶锅', 'desc': '安全耐用，易于清洗', 'price': 149.0}
{'_id': ObjectId('66225d6694ad01cb78a5a33e'), 'name': '告别油腻解放双手，懒人抹布超值囤货装16卷', 'desc': '16卷800节，用完即扔，洗碗打扫不脏手
```



```
{'_id': ObjectId('66226654da5423abc71b70bd'), 'name': '小胖墩单手可拎20寸纯PC拉
查询数据用时: 0.017610549926757812s
开始创建索引
索引创建完成
再次查询数据
{'_id': ObjectId('66225d6694ad01cb78a5a33c'), 'name': '美好即刻呈现富士手机照片打
```



```
{'_id': ObjectId('66226654da5423abc71b70bd'), 'name':
查询数据用时: 0.014007091522216797s
```

由运行结果可知创建索引后查询速度比创建索引前要快，用时更少，但可能由于数据量较少，所以两者之间的差距并不明显。

完整代码如下：

```python
from pymongo import MongoClient
import time

client = MongoClient('mongodb://localhost:27017')
db = client['week7']
collection = db['week7']
def query_data():
    query = collection.find({}, {'name': 1, 'price': 1, 'desc': 1})
    for item in query:
        print(item)
def create_index():
    collection.create_index([('name', 1)])  # 在商品名称字段上创建升序索引
def test_query_speed():
    start = time.time()
    query_data()
    end = time.time()
    print(f"查询数据用时: {end-start}s")
if __name__ == '__main__':
    print("开始查询数据")
    test_query_speed()
    print("开始创建索引")
    create_index()
    print("索引创建完成")
    print("再次查询数据")
    test_query_speed()
```