# 分布式数据库原理与应用实验手册

## 实验五：MongoDB数据库的文档基本操作1：CRUD（一）

|  |
| --- |
| 【实验名称】 实验五：MongoDB数据库的文档基本操作1：CRUD（一） |
| 【实验目的】 |
| 1. 掌握MongoDB CRUD 基本操作 |
| 【实验原理】 |
| 简单、快速浏览MongoDB 中可以执行的CRUD操作。 |
| 【实验环境】 |
| OS：Ubuntu16.04  MongoDB：v3.6 from I9000  DBeaver：v4.1.2 |
| 【实验步骤】  学习mongodb查询文档，完成题目。      find返回的是一个**游标**，通过控制游标实现结果数量展示，例如排序，分页功能    查询语法中文手册：  <https://docs.mongoing.com/mongodb-crud-operations/query-documents>  查询语法中的运算符：  <https://docs.mongoing.com/can-kao/yun-suan-fu>    更新文档中文手册：  <https://docs.mongoing.com/mongodb-crud-operations/update-documents>  更新方法中的运算符：  <https://docs.mongoing.com/can-kao/yun-suan-fu/update-operators> |
| 题目一：插入数据学会使用insertOne、insertMany 或insert 进行插入数据，将以下数据都插入集合： 数据A是一个文档  { item: "notebook2", qty: 50, size: { h: 11, w: 11, uom: "in" }, hit:"hit",tags: ["red", "blank", "blue"], dim\_cm: [ 14, 21 ],status: "C", hit:null, instock: [ { warehouse: "B", qty: 10 } ]}  数据A是一个集合：  [  { item: "journal", qty: 25, size: { h: 14, w: 21, uom: "cm" }, hit:null,status: "A",tags: ["blank", "red"], dim\_cm: [ 14, 21 ],instock: [ { warehouse: "A", qty: 5 }, { warehouse: "C", qty: 15 } ] },  { item: "notebook", qty: 50, size: { h: 8.5, w: 11, uom: "in" }, people:10,tags: ["red", "blank"], dim\_cm: [ 14, 21 ],status: "A" , instock: [ { warehouse: "C", qty: 5 } ]},  { item: "paper", qty: 100, size: { h: 8.5, w: 11, uom: "in" }, tags: ["red", "blank", "plain"], dim\_cm: [ 14, 21 ],status: "D", instock: [ { warehouse: "A", qty: 60 }, { warehouse: "B", qty: 15 } ]},  { item: "planner", qty: 75, size: { h: 22.85, w: 30, uom: "cm" }, tags: ["blank", "red"], dim\_cm: [ 22.85, 30 ], status: "D" , instock: [ { warehouse: "A", qty: 40 }, { warehouse: "B", qty: 5 } ]},  { item: "postcard", qty: 45, size: { h: 10, w: 15.25, uom: "cm" }, tags: ["blue"], dim\_cm: [ 10, 15.25 ], instock: [ { warehouse: "B", qty: 15 }, { warehouse: "C", qty: 35 } ], status: "A" }  ]  ------------------- 题目二：条件查询，学会使用和组合各种查询算子  1. 用count()查询集合的文档数量 2. 单条件查询： status = A 3. 比较算子1 qty 大于等于75的 4. 比较算子2 status不是 A 、C的 5. 逻辑条件1 25 < qty <= 75 6. 逻辑条件2 status = B 或 qty <= 75 7. 数组查询1 tags 有blue 8. 数组查询2 tags 完全包含 plain和 red的 9. 数组查询3 dim\_cm 任一元素x满足 10<=x<15 ($elemMatch) 10. 嵌入文档字段查询1 size文档中 uom = cm 11. 缺失查询1 不包含hit字段的 12. 缺失查询2 hit为空的 13. 综合查询1 tags第一个元素是red的 且 qty 不等于25 db.inventory.find({"tags.0":"red",qty:{$ne:25}}) 14. 综合查询2 tags长度是3 或 status 不是A 15. **综合查询3**  instock中，任一元素x满足 “ 10<x.qty<20 或 x.warehouse=A ”   答案： instock: {$elemMatch:{or:[{qty..},{warehouse..}]} 题目二：查询结果，无查询条件  1. 如果现在有个分页需求，每页2个文档，需要只看第2页数据，在find()结果后使用limit和skip，完成需求。 2. 投影1 只返回item字段 3. 投影2 返回instock中前两个文档，且只包含qty字段   **Reference >mongo shell> cousor > sort()**   1. 排序2 qty升序，size文档h降序 2. 排序3 instock的第0个文档 qty降序  题目三：修改，在简单条件下学会使用updateOne、updateMany配合查询条件更改数据内容  1. 更新1 把status =A的 文档，添加一个 hello=1的 field value 2. 更新2 所有文档 qty值 增长10 3. 更新3 所有文档 dim\_cm中 添加一个数字 20, 并将qty 翻倍 4. 更新4 把status =A的 文档，添加一个aa = {} 的文档，并且aa.k = 10 5. 更新检查 upsert  题目四：删除  1. deleteOne deleteMany |
|  |

## 实验五：MongoDB数据库的文档基本操作1：索引

|  |
| --- |
| 【实验名称】 实验五：MongoDB数据库的文档基本操作2：索引 |
| 【实验目的】 |
| 1. 掌握MongoDB 索引 |
| 【实验原理】 |
| 掌握索引的概念，      全文索引和空间索引。 |
| 【实验环境】 |
| OS：Ubuntu16.04  MongoDB：v3.6 from I9000  DBeaver：v4.1.2 |
| 【实验步骤】 |
| 题目一：建立索引 <https://docs.mongodb.com/v3.4/indexes/>  <https://docs.mongoing.com/indexes>  建立索引createIndex()  db.collection.createIndex( <key and index type specification>, <options> )  <https://docs.mongodb.com/manual/reference/method/db.collection.createIndex/#db.collection.createIndex>  <https://docs.mongoing.com/can-kao/mongo-shell-methods/collection-methods/db-collection-createindex>  索引类型有：  **题目1 创建索引**  单一字段 ： 创建qty索引 , 索引命名为qqq  db.coll.createIndex({qty:1},{name:"qqq"})    createdCollectionAutomatically 意思是： 如果t从没创建出来，创建t的索引以后会 t被创建出来，显示为true  复合索引： 创建 status, tags 索引  db.coll.createIndex({status:1, tags:1})  Multikey索引： 创建size中h的索引  db.coll.createIndex({"size.h":1})  **题目2 查看、删除索引**  db.coll.getIndexes()  .......  db.coll.dropIndex('索引名')  **题目3 查询计划**  db.coll.find({条件}).explain()  如果查询条件使用到索引，winningPlan 会显示相应情况。  **题目二 文本索引和搜索（全文检索）**  简单说：以句子中的单词搜到句子。  插入如下数据  [  { \_id: 1, name: "Java Hut", description: "Coffee and cakes" },  { \_id: 2, name: "Burger Buns", description: "Gourmet hamburgers" },  { \_id: 3, name: "Coffee Shop", description: "Just coffee" },  { \_id: 4, name: "Clothes Clothes Clothes", description: "Discount clothing" },  { \_id: 5, name: "Java Shopping", description: "The following options are available for text indexes only" }  ]  创建索引  <https://docs.mongodb.com/v3.4/text-search/>  <https://docs.mongoing.com/indexes/text-indexes>  db.coll.createIndex({ name: "text", description: "text" } ,{name:"text1"})  注意： 每个集合**只能建一个**文本类型的索引。  文本查询，查询运算符  <https://docs.mongodb.com/v3.4/reference/operator/query/text/>  db.coll.find( { $text: { $search: "java coffee shop" } } )    **如果搜索词是“shop” , \_id: 5 的文档也会被匹配出来了， 因为全文检索会对句子里面的单词进行了去词根的处理。类似 shopping -> shop ; cats -> cat**    空间索引  <https://docs.mongodb.com/v3.4/geospatial-queries/>  <https://docs.mongodb.com/v3.4/reference/geojson/> |
|  |