MongoDB的空间数据存储组织

Contents

1.MongoDB简要介绍

- MongoDB是基于分布式文件存储的非关系型数据库
- 基本思路：从“行”的概念转换成“文档”概念，从而表达复杂的层次。
- 丰富的数据模型：文档的键不预先定义也不会一成不变，不存在模式，也就是所谓的”schema free”。所以当更改数据时比较方便，不需要大量迁移，开发者容易变更数据模型。

1.MongoDB简要介绍

一些概念

•文档：多个键及其有关联的值放置在一起即为文档，例如 {“greeting”:”hello world”}

•集合：一组文档，可以通过名字来标识集合

•无模式：集合中的文档可以是各种各样的

1.MongoDB简要介绍

•扩展性：面向文档的数据模型可实现自动在服务器间分割数据。平衡集群的数据和负载，自动重排文档。

充分考虑性能：

•1.与服务器的交互方式（MongoDB传输协议）

•2.文档动态填充，预分配数据文件

•3.存储时使用内存映射文件，内存管理交由操作系统。

•4.查询优化器会记住“查询最高效的方式”

•5.尽可能将服务器端的处理逻辑交给客户端，设计精简

1.MongoDB简要介绍

MongoDB的功能：

•1.索引，支持通用辅助索引，提供唯一的、复合的地理空间索引

•2.存储JavaScript，在服务端存取JavaScript的函数与值

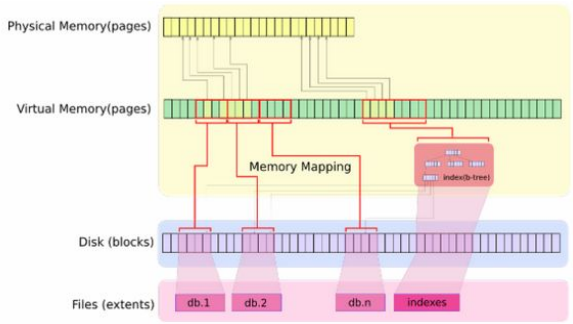
•3.聚合：可以支持MapReduce以及其他的聚合工具

•4.固定集合：集合的大小有限制

•5.文件存储：使用一种比较容易的协议来存储大型文件的元数据

2.MongoDB的数据存储

- 物理存储：Memory-Mapped Files（内存映射文件）



2.MongoDB的数据存储

•MongoDB的存储模型特点

MMF机制的存在，简化了数据访问以及修改的逻辑，读写都交给虚拟内存

•MMF的优点:

不需要自己管理内存和磁盘调度；LRU缓存策略，重启过后，Cache依然存在

- MMF的缺点：

RAM使用会受到磁盘碎片的影响；无法自己对调度算法进行优化，只能使用LRU

3.MongoDB的空间数据支持

- JSON：轻量级的数据交换格式，JavaScript的对象表示法的子集。
- MongoDB的document使用BSON（Binary Json）
- GeoJSON：Json的拓展，一种地理信息数据交换格式，一个对象表示几何、特征或者特征集合。
- 一个GeoJSON总是一个单独的对象组成，其中必须有一个“type”成员

3.MongoDB的空间数据支持

•将空间数据导入到MongoDB时，将Geometry的wkt格式转化为GeoJSON

•格式

```
/**
 * 转换为GeoJSON对象
 */
public  String getGeoJson(String lon,String lat) throws JSONException{
      JSONObject point = new JSONObject();
   point.put("type", "Point");
   JSONArray coord = new JSONArray("["+lon+","+lat+"]");
   point.put("coordinates", coord);
   return point.toString();
}
```

3.MongoDB的空间数据支持

一个表示点对象的document


```
"_id" : ObjectId("4dc82e7f7de36a5ceb000000"),
"PERIMETER" : 0,
"NAME" : "漠河县",
"PYNAME" : "Mohe Xian",
"AREA" : 0,
"ADCODE93" : 232723,
"CNTYPT_ID" : 31,
"CNTYPT_" : 1,
"geom" : {
   "type" : "Point",
   "coordinates" : [
      122.53233,
      52.968872
   ]
},
"ID" : 1031,
"PN" : 1,
"CLASS" : "AI"
```


其中geom类型的数据即为Geometry类型数据

3.MongoDB的空间数据支持

MongoDB索引

1每次查询操作有且只是用一个索引

2.可以使用主键_id创建索引，MongoDB中创建的索引都是二级索引

3.可以在次级文档中创建索引以及嵌入的属性中创建索引。

4.创建复合索引

3.MongoDB的空间数据支持

•地理空间索引机制

•通常数据库使用B+ Tree，如何将地理位置转化为B+ Tree、

•使用GeoHash ，对一张地图进行划分

01 11

00 10

递归划分 使用最终得到的Geohash值作为索引，地图上相近的点转化为具有相同前缀的GeoHash值

GeoHash值得精确度和划分地图的此书澄正比，在MongoDB中默认进行26次划分

3.MongoDB的空间数据支持

使用2D地理空间索引，支持基于位置的数据查询

存储位置数据：

要使用地理所应，必须首先在文档中存储位置数据如经纬度等

loc:[x,y]

创建空间索引：

db.collection.ensureIndex({<location field>:”2d”})

3.MongoDB的空间数据支持

•1.邻域查询

•Db.xqpoint.find({“geom.coordinates”:{$near:[122,52]}})

MongoDB默认返回附近的100个点

也提供了MongoDB的geoNear命令

db.runCommand({geoNear:"xqpoint",near:[122,56],num:2})

3.MongoDB的空间数据支持

2.范围查询

$within操作符 支持三种形状（矩形、圆形、多边形）

例如：

```
b.xqpoint.find({"geom.coordinates":{$within:{$box:box}}})
db.xqpoint.find({"geom.coordinates":{$within:{$center:[center,radius]}}})
db.xqpoint.find({"geom.coordinates":{$within:{$polygon:polygon1}}})
```