《大数据基础编程、实验和案例教程（第2版）》

教材官网：
http://dblab.xmu.edu.cn/post/bigdatapractice2/

温馨提示：编辑幻灯片母版，可以修改每页PPT的厦大校徽和底部文字

第6章 典型NoSQL数据库的安装和使用

（PPT版本号：2020年12月版本）

扫一扫访问教材官网

林子雨

厦门大学计算机科学系

E-mail: ziyulin@xmu.edu.cn

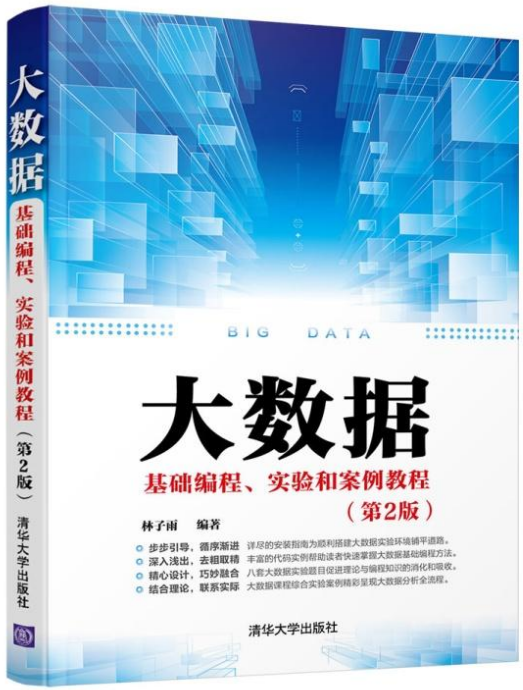
主页：http://dblab.xmu.edu.cn/linziyu

教材简介

本书是与《大数据技术原理与应用（第3版）》教材配套的唯一指定实验指导书

林子雨编著《大数据基础编程、实验和案例教程（第2版）》
清华大学出版社 ISBN:978-7-302-55977-1 定价：69元，2020年10月第2版
教材官网： http://dblab.xmu.edu.cn/post/bigdatapractice2/

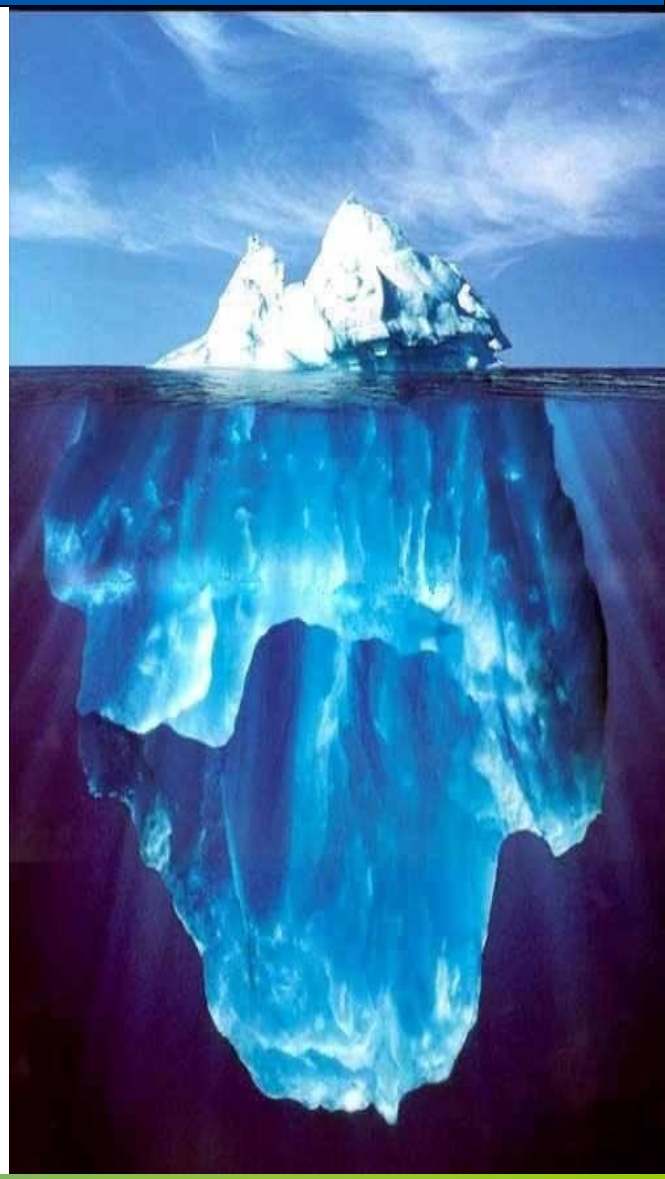
扫一扫访问
教材官网



- 步步引导，循序渐进，详尽的安装指南为顺利搭建大数据实验环境铺平道路
- 深入浅出，去粗取精，丰富的代码实例帮助快速掌握大数据基础编程方法
- 精心设计，巧妙融合，八套大数据实验题目促进理论与编程知识的消化和吸收
- 结合理论，联系实际，大数据课程综合实验案例精彩呈现大数据分析全流程

提纲

6.1 Redis安装和使用
6.2 MongoDB的安装和使用



高校大数据课程
公 共 服 务 平 台

百度搜索厦门大学数据库实验室网站访问平台

6.1 Redis安装和使用

6.1.1 Redis简介
6.1.2 安装Redis
6.1.3 Redis实例演示

6.1.1 Redis简介

Redis是一个键值（key-value）存储系统，即键值对非关系型数据库，和Memcached类似，目前正在被越来越多的互联网公司采用。Redis作为一个高性能的键值数据库，不仅在很大程度上弥补了memcached这类键值存储的不足，而且在部分场合下可以对关系数据库起到很好的补充作用。Redis提供了Python、Ruby、Erlang、PHP客户端，使用很方便。

Redis支持存储的值（value）类型包括string(字符串)、list(链表)、set(集合)和zset(有序集合)。这些数据类型都支持push/pop、add/remove以及取交集、并集和差集等丰富的操作，而且这些操作都是原子性的。在此基础上，Redis支持各种不同方式的排序。与memcached一样，为了保证效率，Redis中的数据都是缓存在内存中的，它会周期性地把更新的数据写入磁盘，或者把修改操作写入追加的记录文件；此外，Redis还实现了主从（master-slave）同步。

6.1.2 安装Redis

访问Redis官网（http://www.redis.cn/）下载安装包redis-5.0.5.tar.gz

执行以下命令将Redis解压至“/usr/local/”目录下并重命名：

```
$ cd ~
$ sudo tar -zxvf ./下载/redis-5.0.5.tar.gz -C /usr/local
$ cd /usr/local
$ sudo mv ./redis-5.0.5 ./redis
```

然后，执行如下命令把redis目录的权限赋予hadoop用户：

```
$ sudo chown -R hadoop:hadoop ./redis
```

接下来，进入“/usr/local/redis”目录，输入以下命令编译和安装Redis：

```
$ sudo make
$ sudo make install
```

6.1.2 安装Redis

至此，Redis已经安装完成，现在可以执行如下命令开启Redis服务器：

```
$ cd /usr/local/redis
$ ./src/redis-server
```

6.1.2 安装Redis

然后，再新建一个终端，输入如下命令启动Redis客户端：

```
$ cd /usr/local/redis
$ ./src/redis-cli
```

如图6-2所示，客户端连上服务器之后，会显示“127.0.0.1:6379>”的命令提示符信息，表示服务器的IP地址为127.0.0.1，端口为6379。现在可以执行简单的操作，比如，设置键为”hello”，值为”world”，并且取出键为”hello”时对应的值，图6-2给出了具体的操作效果。

```
hadoop@ubuntu:/usr/local/redis$  ./src/redis-cli
127.0.0.1:6379> set hello world
OK
127.0.0.1:6379> get hello
"world"
```

6.1.3 Redis实例演示

假设有三个表，即Student、Course和SC，三个表的字段（列）和数据如图6-3所示。

Sno	Sname	Ssex	Sage	Sdept
95001	李勇	男	22	CS
95002	刘晨	女	19	IS
95003	王敏	女	18	MA
95004	张立	男	19	IS

Cno	Cname	Credit
1	数据库	4
2	数学	2
3	信息系统	4
4	操作系统	3
5	数据结构	4
6	数据处理	2
7	PASCAL语言	4

Sno	Cno	Grade
95001	1	92
95001	2	85
95001	3	88
95002	2	90
95002	3	80

6.1.3 Redis实例演示

Redis数据库是以<key,value>的形式存储数据，把三个表的数据存入Redis数据库时，key和value的确定方法如下：

key=表名:主键值:列名
value=列值

例如，把每个表的第一行记录保存到Redis数据中，需要执行的命令和执行结果如图6-4所示。set SC:95001:1:Grade 92

```
127.0.0.1:6379> set Student:95001:Sname 李勇
OK
127.0.0.1:6379> set Course:1:Cname 数据库
OK
127.0.0.1:6379> set SC:95001:1:Grade 92
OK
```

6.1.3 Redis实例演示

1. 插入数据

```
127.0.0.1:6379> set Course:8:Cname 算法
OK
127.0.0.1:6379> set Course:8:Ccredit 4
OK
127.0.0.1:6379>
```

6.1.3 Redis实例演示

2. 修改数据

```
127.0.0.1:6379> get Course:8:Cname
算法
127.0.0.1:6379> set Course:8:Cname 编译原理
OK
127.0.0.1:6379> get Course:8:Cname
编译原理
127.0.0.1:6379>
```

6.1.3 Redis实例演示

3. 删除数据

```
127.0.0.1:6379> get Course:8:Cname
编译原理
127.0.0.1:6379> del Course:8:Cname
1
127.0.0.1:6379> get Course:8:Cname

127.0.0.1:6379>
```

6.2 MongoDB的安装和使用

6.2.1 MongDB简介
6.2.2 安装MongoDB
6.2.3使用Shell命令操作MongoDB
6.2.4 Java API编程实例

6.2.1 MongDB简介

MongoDB是一个基于分布式文件存储的文档数据库，介于关系数据库和非关系数据库之间，是非关系数据库当中功能最丰富、最像关系数据库的一种NoSQL数据库。MongoDB支持的数据结构非常松散，是类似json的bson格式，因此可以存储比较复杂的数据类型。Mongo最大的特点是支持的查询语言非常强大，语法有点类似于面向对象的查询语言，几乎可以实现类似关系数据库单表查询的绝大部分功能，而且还支持对数据建立索引。

6.2.2 安装MongoDB

MongoDB既可以安装在Windows系统下使用，也可以安装在Linux系统下使用，这里采用Linux系统。MongoDB安装很简单，无需下载源文件，可以直接用apt-get命令进行安装。

但是，需要说明的是，如果直接使用“sudo apt-get install mongodb”命令进行安装，默认安装的版本是MongoDB 2.6.10。由于目前MongoDB已经升级到4.0.16，这里将通过添加软件源的方式来安装4.0.16版本。

首先，在Linux系统中打开一个终端，执行如下命令导入公共秘钥到包管理器中：

```
$ sudo apt-key adv --keyserver hkp://keyserver.ubuntu.com:80
--recv 9DA31620334BD75D9DCB49F368818C72E52529D4
```

6.2.2 安装MongoDB

然后，创建MongoDB的文件列表，命令如下：

```
#对于Ubuntu18.04，使用如下命令:
$ echo "deb [ arch=amd64 ] https://repo.mongodb.org/apt/ubuntu
bionic/mongodb-org/4.0 multiverse" | sudo tee
/etc/apt/sources.list.d/mongodb.list
#对于Ubuntu16.04，使用如下命令：
$ echo "deb [ arch=amd64,arm64 ] https://repo.mongodb.org/apt/ubuntu
xenial/mongodb-org/4.0 multiverse" | sudo tee
/etc/apt/sources.list.d/mongodb.list
```

6.2.2 安装MongoDB

执行如下命令来更新包管理器：

```
$ sudo apt-get update
```

最后，执行如下命令安装MongoDB：

```
$ sudo apt install mongodb-org
```

安装完成后，在终端输入以下命令查看MongoDB版本：

```
$ mongo -version
```

6.2.2 安装MongoDB

如果能够输出版本信息（如图6-8所示），则表明安装成功。

```
cjx@ubuntu18:~$ mongo -version
MongoDB shell version v4.0.16
git version: 2a5433168a53044cb6b4fa8083e4cfd7ba142221
OpenSSL version: OpenSSL 1.1.1  11 Sep 2018
allocator: tcmalloc
modules: none
build environment:
    distmod: ubuntu1804
    distarch: x86_64
    target_arch: x86_64
```

安装成功以后，启动MongoDB的命令如下：

```
$ sudo service mongod start
```

默认设置下，MongoDB是随Ubuntu启动而自动启动的。可以输入以下命令查看是否启动成功：

```
$ pgrep mongod -l   #注意：-l是英文字母l，不是阿拉伯数字1
```

6.2.2 安装MongoDB

如果能够出现如图6-9所示信息，则表明启动成功：

```
cjx@ubuntu18:~$ pgrep mongod -l
2988 mongod
```

使用MongoDB结束后，关闭MongoDB的命令如下：

```
$ sudo service mongod stop
```

6.2.3使用Shell命令操作MongoDB

1. 进入MongoDB Shell模式

在Linux系统打开一个终端，输入如下命令启动MongoDB：

```
$ sudo service mongod start
```

再输入如下命令进入MongoDB Shell模式：

执行该命令后，屏幕截图如图6-10所示。

```
2020-02-20T10:32:00.172+0800 I STORAGE  [initandlisten] **          See http://dochub.mongodb.org/core/prodn
otes-filesystem
2020-02-20T10:32:01.188+0800 I CONTROL  [initandlisten]
2020-02-20T10:32:01.188+0800 I CONTROL  [initandlisten] ** WARNING: Access control is not enabled for the da
tabase.
2020-02-20T10:32:01.188+0800 I CONTROL  [initandlisten] **          Read and write access to data and config
uration is unrestricted.
2020-02-20T10:32:01.188+0800 I CONTROL  [initandlisten]
---
Enable MongoDB's free cloud-based monitoring service, which will then receive and display
metrics about your deployment (disk utilization, CPU, operation statistics, etc).

The monitoring data will be available on a MongoDB website with a unique URL accessible to you
and anyone you share the URL with. MongoDB may use this information to make product
improvements and to suggest MongoDB products and deployment options to you.

To enable free monitoring, run the following command: db.enableFreeMonitoring()
To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
---

>
```

6.2.3使用Shell命令操作MongoDB

2. 常用操作命令

```
> show dbs
admin   0.000GB
config  0.000GB
local   0.000GB
> use School
switched to db School
> db.createCollection('teacher')
{ "ok" : 1 }
> show dbs
School  0.000GB
admin   0.000GB
config  0.000GB
local   0.000GB
>
```

6.2.3使用Shell命令操作MongoDB

3.简单操作演示

（1）切换到School数据库

命令如下：

```
> use School
```

（2）创建集合

创建集合（Collection）的命令如下：

```
> use School
switched to db School
> show collections
> db.createCollection('teacher')
{ "ok" : 1 }
> show collections
teacher
```

6.2.3使用Shell命令操作MongoDB

（3）插入数据

```
> db.student.insert({_id:1,sname:'zhangsan',sage:20})
WriteResult({ "nInserted" : 1 })
> db.student.find()
{ "_id" : 1, "sname" : "zhangsan", "sage" : 20 }
> db.student.save({_id:1,sname:'zhangsan',sage:22})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.student.find()
{ "_id" : 1, "sname" : "zhangsan", "sage" : 22 }
> db.student.insert({_id:1,sname:'zhangsan',sage:25})
WriteResult({
        "nInserted" : 0,
        "writeError" : {
                "code" : 11000,
                "errmsg" : "E11000 duplicate key error collection: School.student index: _id_ dup key: { : 1.0 }"
        }
})
> db.student.find()
{ "_id" : 1, "sname" : "zhangsan", "sage" : 22 }
>
```

insert：插入成功

save：_id相同，更新数据

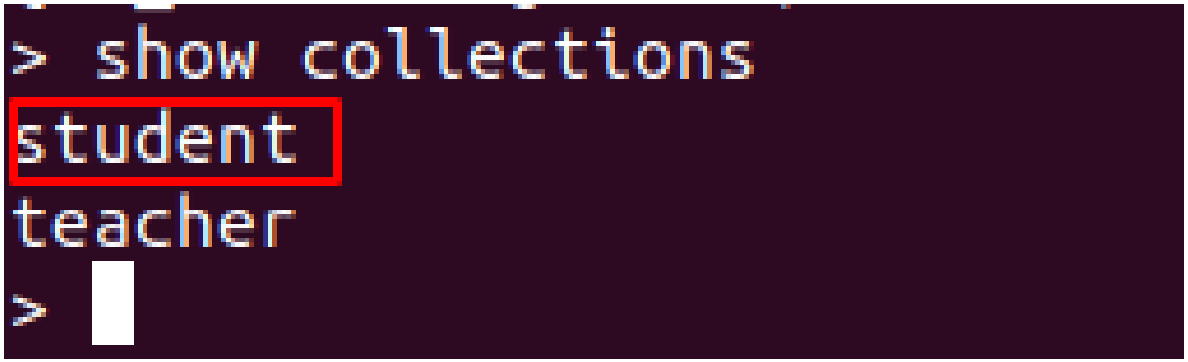
insert: _id相同，插入失败，不做操作

6.2.3使用Shell命令操作MongoDB

```
> s = [{sname:'lisi',sage:20},{sname:'wangwu',sage:20},{sname:'chenliu',sage:20}]
[
        {
                "sname" : "lisi",
                "sage" : 20
        },
        {
                "sname" : "wangwu",
                "sage" : 20
        },
        {
                "sname" : "chenliu",
                "sage" : 20
        }
]
> db.student.insert(s)
BulkWriteResult({
        "writeErrors" : [ ],
        "writeConcernErrors" : [ ],
        "nInserted" : 3,
        "nUpserted" : 0,
        "nMatched" : 0,
        "nModified" : 0,
        "nRemoved" : 0,
        "upserted" : [ ]
})
> db.student.find()
{ "_id" : 1, "sname" : "zhangsan", "sage" : 22 }
{ "_id" : ObjectId("5e4df819610e7a10d6808e2f"), "sname" : "lisi", "sage" : 20 }
{ "_id" : ObjectId("5e4df819610e7a10d6808e30"), "sname" : "wangwu", "sage" : 20 }
{ "_id" : ObjectId("5e4df819610e7a10d6808e31"), "sname" : "chenliu", "sage" : 20 }
>
```

6.2.3使用Shell命令操作MongoDB

运行完以上例子，student已自动创建，这也说明MongoDB不需要预先定义集合（collection），在第一次插入数据后，集合会被自动创建。此时，可以使用“show collections”命令查询数据中当前已经存在的集合，如图6-15所示。



6.2.3使用Shell命令操作MongoDB

（4）查找数据

查找数据所使用的基本命令格式如下：

```
> db.youCollection.find(criteria, filterDisplay)
```

查询所有记录

```
> db.student.find()
```

查询sname='lisi'的记录

```
> db.student.find({sname: 'lisi'})
```

查询指定列sname、sage数据

```
> db.student.find({},{sname:1, sage:1})
```

6.2.3使用Shell命令操作MongoDB

AND条件查询

```
> db.student.find({sname: 'zhangsan', sage: 22})
```

OR条件查询

```
> db.student.find({$or: [{sage: 22}, {sage: 25}]})
```

格式化输出

```
> db.student.find().pretty()
{ "_id" : 1, "sname" : "zhangsan", "sage" : 22 }
{
        "_id" : ObjectId("5e4df819610e7a10d6808e2f"),
        "sname" : "lisi",
        "sage" : 20
}
{
        "_id" : ObjectId("5e4df819610e7a10d6808e30"),
        "sname" : "wangwu",
        "sage" : 20
}
{
        "_id" : ObjectId("5e4df819610e7a10d6808e31"),
        "sname" : "chenliu",
        "sage" : 20
}
>
```

6.2.3使用Shell命令操作MongoDB

（5）修改数据

修改数据的基本命令格式如下：

```
> db.youCollection.update(criteria, objNew, upsert, multi )
```

这里给出一个实例，语句如下：

```
> db.student.update({sname: 'lisi'}, {$set: {sage: 30}}, false, true)
```

```
> db.student.update({sname:'lisi'},{$set:{sage:30}},false,true)
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.student.find({sname:'lisi'})
{ "_id" : ObjectId("5e4df819610e7a10d6808e2f"), "sname" : "lisi", "sage" : 30 }
>
```

6.2.3使用Shell命令操作MongoDB

（6）删除数据

```
> db.student.remove({sname:'chenliu'})
WriteResult({ "nRemoved" : 1 })
> db.student.find()
{ "_id" : 1, "sname" : "zhangsan", "sage" : 22 }
{ "_id" : ObjectId("5e4df819610e7a10d6808e2f"), "sname" : "lisi", "sage" : 30 }
{ "_id" : ObjectId("5e4df819610e7a10d6808e30"), "sname" : "wangwu", "sage" : 20 }
>
```

（7）删除集合

```
> show collections
student
teacher
> db.student.drop()
true
> show collections
teacher
>
```

4. 退出MongoDB Shell模式

可以输入如下命令退出MongoDB Shell模式：

```
> exit
```

6.2.4 Java API编程实例

编写Java程序访问MongoDB数据库时，首先，需要下载Java MongoDB Driver驱动JAR包，Java MongoDB Driver下载地址如下：

https://repo1.maven.org/maven2/org/mongodb/mongo-java-driver/3.12.1/mongo-java-driver-3.12.1.jar

6.2.4 Java API编程实例

```
import java.util.ArrayList;
import java.util.List;
import org.bson.Document;
import com.mongodb.MongoClient;
import com.mongodb.client.MongoCollection;
import com.mongodb.client.MongoCursor;
import com.mongodb.client.MongoDatabase;
import com.mongodb.client.model.Filters;
public class TestMongoDB {
    /**
     * @param args
     */
    public static void main(String[] args) {
//      insert();//插入数据。执行插入时，可将其他三句函数调用语句注释，下同
        find(); //查找数据
//      update();//更新数据
//      delete();//删除数据
    }
```

6.2.4 Java API编程实例

```
/**
     * 返回指定数据库中的指定集合
     * @param dbname 数据库名
     * @param collectionname 集合名
     * @return
     */
    //MongoDB无需预定义数据库和集合,在使用的时候会自动创建
    public static MongoCollection<Document> getCollection(String
dbname,String collectionname){
        //实例化一个mongo客户端,服务器地址：localhost(本地)，端口号：27017
        MongoClient  mongoClient=new MongoClient("localhost",27017);
        //实例化一个mongo数据库
        MongoDatabase mongoDatabase = mongoClient.getDatabase(dbname);
        //获取数据库中某个集合
        MongoCollection<Document> collection =
mongoDatabase.getCollection(collectionname);
        return collection;
    }
```

6.2.4 Java API编程实例

```
/**
    * 插入数据
    */
   public static void insert(){
       try{
           //连接MongoDB，指定连接数据库名，指定连接表名。
           MongoCollection<Document> collection= getCollection("School","student");   //数据库
名:School 集合名:student
           //实例化一个文档,文档内容为{sname:'Mary',sage:25}，如果还有其他字段，可以继续追加
append
           Document doc1=new Document("sname","Mary").append("sage", 25);
           //实例化一个文档,文档内容为{sname:'Bob',sage:20}
           Document doc2=new Document("sname","Bob").append("sage", 20);
           List<Document> documents = new ArrayList<Document>();
           //将doc1、doc2加入到documents列表中
           documents.add(doc1);
           documents.add(doc2);
           //将documents插入集合
           collection.insertMany(documents);
           System.out.println("插入成功");
       }catch(Exception e){
           System.err.println( e.getClass().getName() + ": " + e.getMessage() );
       }
   }
```

6.2.4 Java API编程实例

```
/**
    * 查询数据
    */
   public static void find(){
      try{
         MongoCollection<Document> collection = getCollection("School","student");  //
数据库名:School 集合名:student
         //通过游标遍历检索出的文档集合
//         MongoCursor<Document>  cursor= collection.find(new
Document("sname","Mary")). projection(new
Document("sname",1).append("sage",1).append("_id", 0)).iterator();   //find查询条件：
sname='Mary'。projection筛选：显示sname和sage,不显示_id(_id默认会显示)
         //查询所有数据
         MongoCursor<Document>  cursor= collection.find().iterator();
         while(cursor.hasNext()){
            System.out.println(cursor.next().toJson());
         }
      }catch(Exception e){
         System.err.println( e.getClass().getName() + ": " + e.getMessage() );
      }
   }
```

6.2.4 Java API编程实例

```
/**
    * 更新数据
    */
   public static void update(){
       try{
           MongoCollection<Document> collection =
getCollection("School","student");  //数据库名:School 集合名:student
            //更新文档，将文档中sname='Mary'的文档修改为sage=22
            collection.updateMany(Filters.eq("sname", "Mary"), new
Document("$set",new Document("sage",22)));
            System.out.println("更新成功！ ");
       }catch(Exception e){
           System.err.println( e.getClass().getName() + ": " + e.getMessage() );
       }
   }
```

6.2.4 Java API编程实例

```
/**
    * 删除数据
    */
   public static void delete(){
      try{
         MongoCollection<Document> collection =
getCollection("School","student");  //数据库名:School 集合名:student
            //删除符合条件的第一个文档
         collection.deleteOne(Filters.eq("sname", "Bob"));
         //删除所有符合条件的文档
         //collection.deleteMany (Filters.eq("sname", "Bob"));
         System.out.println("删除成功！ ");
      }catch(Exception e){
         System.err.println( e.getClass().getName() + ": " + e.getMessage() );
      }
   }
}
```

6.2.4 Java API编程实例

每次在Eclipse中执行完该程序，都可以在Linux系统的MongoDB Shell模式下查看结果。比如，在Eclipse执行完更新操作后，在MongoDB Shell模式下输入命令“db.student.find()”，就可以查看student集合的所有数据（如图6-20所示）。

```
> db.student.find()  更新操作前
{ "_id" : ObjectId("5e4f48909608324246 34ed69"), "sname" : "Mary", "sage" : 25 }
{ "_id" : ObjectId("5e4f48909608324246 34ed6a"), "sname" : "Bob", "sage" : 20 }
> db.student.find()  更新操作后
{ "_id" : ObjectId("5e4f48909608324246 34ed69"), "sname" : "Mary", "sage" : 22 }
{ "_id" : ObjectId("5e4f48909608324246 34ed6a"), "sname" : "Bob", "sage" : 20 }
>
```

6.3 本章小结

传统的关系数据库可以较好地支持结构化数据存储和管理，但是，Web 2.0的迅猛发展以及大数据时代的到来，使关系数据库的发展越来越力不从心。在大数据时代，数据类型繁多，包括结构化数据和各种非结构化数据，其中，非结构化数据的比例更是高达90%以上。因此，在新的应用需求驱动下，各种新型的NoSQL数据库不断涌现，并逐渐获得市场的青睐。NoSQL数据库主要包括键值数据库、列族数据库、文档数据库和图数据库等4种类型，前面在第5章介绍的HBase就属于列族数据库。

键值数据库和文档数据库是两种应用比较广泛的NoSQL数据库，因此，本章选取了两个具有代表性的产品进行介绍，包括键值数据库Redis和文档数据库MongoDB，详细介绍了这两种数据库的安装和使用方法，并给出了编程实例。

附录A：主讲教师林子雨简介

主讲教师：林子雨

单位：厦门大学计算机科学系
E-mail: ziyulin@xmu.edu.cn
个人网页：http://dblab.xmu.edu.cn/post/linziyu
数据库实验室网站：http://dblab.xmu.edu.cn

扫一扫访问个人主页

林子雨，男，1978年出生，博士（毕业于北京大学），全国高校知名大数据教师，现为厦门大学计算机科学系副教授，曾任厦门大学信息科学与技术学院院长助理、晋江市发展和改革局副局长。中国计算机学会数据库专业委员会委员，中国计算机学会信息系统专业委员会委员。国内高校首个“数字教师”提出者和建设者，厦门大学数据库实验室负责人，厦门大学云计算与大数据研究中心主要建设者和骨干成员，2013年度、2017年度和2020年度厦门大学教学类奖教金获得者，荣获2019年福建省精品在线开放课程、2018年厦门大学高等教育成果特等奖、2018年福建省高等教育教学成果二等奖、2018年国家精品在线开放课程。主要研究方向为数据库、数据仓库、数据挖掘、大数据、云计算和物联网，并以第一作者身份在《软件学报》《计算机学报》和《计算机研究与发展》等国家重点期刊以及国际学术会议上发表多篇学术论文。作为项目负责人主持的科研项目包括1项国家自然科学青年基金项目(No.61303004)、1项福建省自然科学青年基金项目(No.2013J05099)和1项中央高校基本科研业务费项目(No.2011121049)，主持的教改课题包括1项2016年福建省教改课题和1项2016年教育部产学协作育人项目，同时，作为课题负责人完成了国家发改委城市信息化重大课题、国家物联网重大应用示范工程区域试点泉州市工作方案、2015泉州市互联网经济调研等课题。中国高校首个“数字教师”提出者和建设者，2009年至今，“数字教师”大平台累计向网络免费发布超过1000万字高价值的研究和教学资料，累计网络访问量超过1000万次。打造了中国高校大数据教学知名品牌，编著出版了中国高校第一本系统介绍大数据知识的专业教材《大数据技术原理与应用》，并成为京东、当当网等网店畅销书籍；建设了国内高校首个大数据课程公共服务平台，为教师教学和学生学习大数据课程提供全方位、一站式服务，年访问量超过200万次，累计访问量超过1000万次。

附录B：大数据学习路线图

大数据学习路线图访问地址：http://dblab.xmu.edu.cn/post/10164/

附录C：林子雨大数据系列教材

了解全部教材信息：http://dblab.xmu.edu.cn/post/bigdatabook/

附录D：《大数据导论（通识课版）》教材

开设全校公共选修课的优质教材

本课程旨在实现以下几个培养目标：

- 引导学生步入大数据时代，积极投身大数据的变革浪潮之中
- 了解大数据概念，培养大数据思维，养成数据安全意识
- 认识大数据伦理，努力使自己的行为符合大数据伦理规范要求
- 熟悉大数据应用，探寻大数据与自己专业的应用结合点
- 激发学生基于大数据的创新创业热情

高等教育出版社 ISBN:978-7-04-053577-8 定价：32元

教材官网：http://dblab.xmu.edu.cn/post/bigdataintroduction/

附录E：《大数据导论》教材

- 林子雨 编著《大数据导论》
- 人民邮电出版社，2020年9月第1版
- ISBN:978-7-115-54446-9 定价：49.80元

教材官网：http://dblab.xmu.edu.cn/post/bigdata-introduction/

开设大数据专业导论课的优质教材

扫一扫访问教材官网

附录F：《大数据技术原理与应用》教材

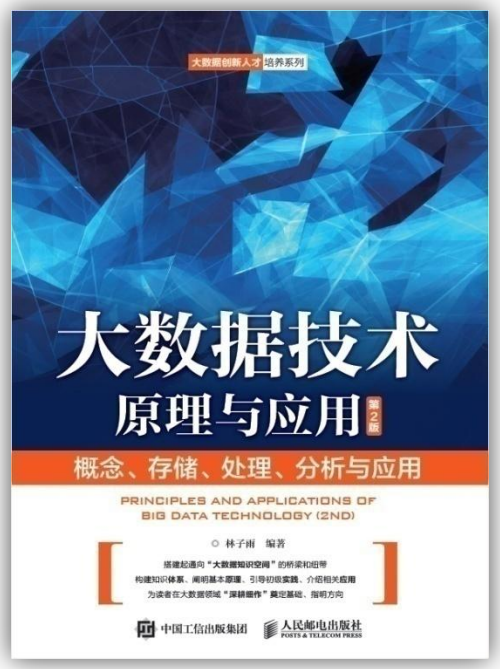
《大数据技术原理与应用——概念、存储、处理、分析与应用（第2版）》，由厦门大学计算机科学系林子雨博士编著，是国内高校第一本系统介绍大数据知识的专业教材。人民邮电出版社 ISBN:978-7-115-44330-4 定价：49.80元

全书共有15章，系统地论述了大数据的基本概念、大数据处理架构Hadoop、分布式文件系统HDFS、分布式数据 库HBase、NoSQL数据库、云数据库、分布式并行编程模型MapReduce、Spark、流计算、图计算、数据可视化以及大数据在互联网、生物医学和物流等各个领域的应用。在Hadoop、HDFS、HBase和MapReduce等重要章节，安排了入门级的实践操作，让读者更好地学习和掌握大数据关键技术。

本书可以作为高等院校计算机专业、信息管理等相关专业的大数据课程教材，也可供相关技术人员参考、学习、培训之用。

欢迎访问《大数据技术原理与应用——概念、存储、处理、分析与应用》教材官方网站：
http://dblab.xmu.edu.cn/post/bigdata

扫一扫访问教材官网



附录G：《大数据基础编程、实验和案例教程（第2版）》

本书是与《大数据技术原理与应用（第3版）》教材配套的唯一指定实验指导书

大数据教材

1+1黄金组合

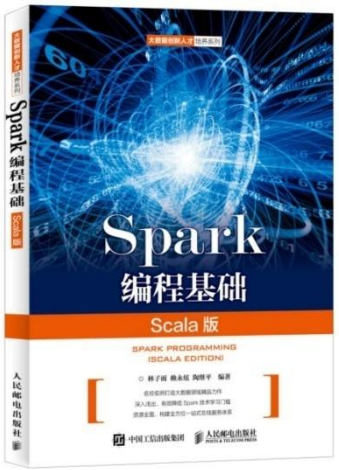
厦门大学林子雨编著

配套实验指导书

- 步步引导，循序渐进，详尽的安装指南为顺利搭建大数据实验环境铺平道路
- 深入浅出，去粗取精，丰富的代码实例帮助快速掌握大数据基础编程方法
- 精心设计，巧妙融合，八套大数据实验题目促进理论与编程知识的消化和吸收
- 结合理论，联系实际，大数据课程综合实验案例精彩呈现大数据分析全流程

林子雨编著《大数据基础编程、实验和案例教程（第2版）》
清华大学出版社 ISBN:978-7-302-55977-1 定价：69元 2020年10月第2版

附录H：《Spark编程基础（Scala版）》



《Spark编程基础（Scala版）》

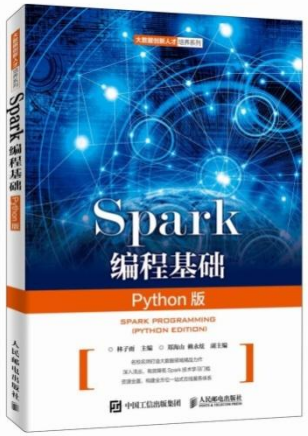
厦门大学 林子雨，赖永炫，陶继平 编著

披荆斩棘，在大数据丛林中开辟学习捷径
填沟削坎，为快速学习Spark技术铺平道路
深入浅出，有效降低Spark技术学习门槛
资源全面，构建全方位一站式在线服务体系

人民邮电出版社出版发行，ISBN:978-7-115-48816-9
教材官网：http://dblab.xmu.edu.cn/post/spark/

本书以Scala作为开发Spark应用程序的编程语言，系统介绍了Spark编程的基础知识。全书共8章，内容包括大数据技术概述、Scala语言基础、Spark的设计与运行原理、Spark环境搭建和使用方法、RDD编程、Spark SQL、Spark Streaming、Spark MLlib等。本书每个章节都安排了入门级的编程实践操作，以便读者更好地学习和掌握Spark编程方法。本书官网免费提供了全套的在线教学资源，包括讲义PPT、习题、源代码、软件、数据集、授课视频、上机实验指南等。

附录I：《Spark编程基础（Python版）》



《Spark编程基础（Python版）》

厦门大学 林子雨，郑海山，赖永炫 编著

披荆斩棘，在大数据丛林中开辟学习捷径
填沟削坎，为快速学习Spark技术铺平道路
深入浅出，有效降低Spark技术学习门槛
资源全面，构建全方位一站式在线服务体系

人民邮电出版社出版发行，ISBN:978-7-115-52439-3
教材官网： http://dblab.xmu.edu.cn/post/spark-python/

本书以Python作为开发Spark应用程序的编程语言，系统介绍了Spark编程的基础知识。全书共8章，内容包括大数据技术概述、Spark的设计与运行原理、Spark环境搭建和使用方法、RDD编程、Spark SQL、Spark Streaming、Structured Streaming、Spark MLlib等。本书每个章节都安排了入门级的编程实践操作，以便读者更好地学习和掌握Spark编程方法。本书官网免费提供了全套的在线教学资源，包括讲义PPT、习题、源代码、软件、数据集、上机实验指南等。

附录J：高校大数据课程公共服务平台

高校大数据课程

公 共 服 务 平 台

http://dblab.xmu.edu.cn/post/bigdata-teaching-platform/

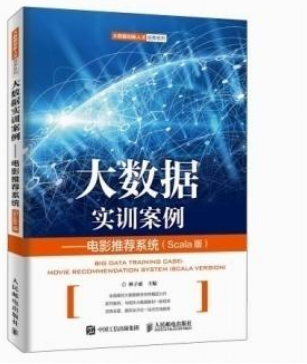
扫一扫访问平台主页

扫一扫观看3分钟FLASH动画宣传片

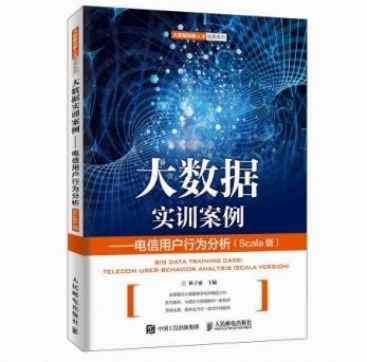
附录K：高校大数据实训课程系列案例教材

为了更好满足高校开设大数据实训课程的教材需求，厦门大学数据库实验室林子雨老师团队联合企业共同开发了《高校大数据实训课程系列案例》，目前已经完成开发的系列案例包括：

《电影推荐系统》（已经于2019年5月出版）
《电信用户行为分析》（已经于2019年5月出版）
《实时日志流处理分析》
《微博用户情感分析》
《互联网广告预测分析》
《网站日志处理分析》



系列案例教材将于2019年陆续出版发行，教材相关信息，敬请关注网页后续更新！
http://dblab.xmu.edu.cn/post/shixunkecheng/



扫一扫访问大数据实训课程系列案例教材主页


Thank You!