2017年1月4日 星期三 22:34

1. 导入hbase的依赖包
2. 配置HBaseConfiguration

```
@Before
public void init() throws Exception {
    config = HBaseConfiguration.create();// 配置
    config.set("hbase.zookeeper.quorum", "master1ha,master2,master2ha");// zookeeper地址
    config.set("hbase.zookeeper.property.clientPort", "2181");// zookeeper端口
    connection = ConnectionFactory.createConnection(config);
    table = connection.getTable(TableName.valueOf("user"));
}
```

创建表

```
/**
 * 创建一个表
 *
 * @throws Exception
 */
@Test
public void createTable() throws Exception {
    // 创建表管理类
    HBaseAdmin admin = new HBaseAdmin(config); // hbase表管理
    // 创建表描述类
    TableName tableName = TableName.valueOf("test3"); // 表名称
    HTableDescriptor desc = new HTableDescriptor(tableName);
    // 创建列族的描述类
    HColumnDescriptor family = new HColumnDescriptor("info"); // 列族
    // 将列族添加到表中
    desc.addFamily(family);
    HColumnDescriptor family2 = new HColumnDescriptor("info2"); // 列族
    // 将列族添加到表中
    desc.addFamily(family2);
    // 创建表
    admin.createTable(desc); // 创建表
}
```

增加数据

```
/**
 * 向hbase中增加数据
 *
 * @throws Exception
 */
@SuppressWarnings({ "deprecation", "resource" })
@Test
public void insertData() throws Exception {
    //创建的数据封装类
    Put put = new Put(Bytes.toBytes("wangsenfeng 1234"));
    put.add(Bytes.toBytes("info1"), Bytes.toBytes("name"), Bytes.toBytes("zhangsan"));
    put.add(Bytes.toBytes("info1"), Bytes.toBytes("age"), Bytes.toBytes(23));
    put.add(Bytes.toBytes("info1"), Bytes.toBytes("sex"), Bytes.toBytes(0));
    put.add(Bytes.toBytes("info1"), Bytes.toBytes("address"), Bytes.toBytes("safadsfads"));
    //添加数据
    table.put(put);
}
```

rowkey一样的话就是覆盖，就是修改

删除数据

```java
@Test
public void deleteDate() throws Exception {
    //创建删除的rowkey
    Delete delete = new Delete(Bytes.toBytes("wangsenfeng_200"));
    //删除操作
    table.delete(delete);

}
```

```java
@Test
public void deleteDate() throws Exception {
    //创建删除的rowkey
    Delete delete = new Delete(Bytes.toBytes("wangsenfeng_2000"));
    delete.addColumn(Bytes.toBytes("info2"), Bytes.toBytes("name"));
    //删除操作
    table.delete(delete);

}
```

查询三种方法：

单条查询

```java
@Test
public void queryData() throws Exception {
    //创建封装查询条件的类
    Get get = new Get(Bytes.toBytes("wangsenfeng_200"));
    //创建查询
    Result result = table.get(get);
    byte[] value = result.getValue(Bytes.toBytes("info2"), Bytes.toBytes("name"));
    System.out.println(Bytes.toString(value));

}
```

全表扫描（scan，结果ResultScanner可以使用迭代器）

```java
@Test
public void scanData() throws Exception {
    //设置全表扫描封装类
    Scan scan = new Scan();
    //扫描
    ResultScanner scanner = table.getScanner(scan);
    for (Result result : scanner) {
        byte[] value = result.getValue(Bytes.toBytes("info2"), Bytes.toBytes("name"));
        byte[] sex = result.getValue(Bytes.toBytes("info2"), Bytes.toBytes("sex"));
        byte[] address = result.getValue(Bytes.toBytes("info2"), Bytes.toBytes("address"));
        byte[] age = result.getValue(Bytes.toBytes("info2"), Bytes.toBytes("age"));
        System.out.println(Bytes.toString(value));
        System.out.println(Bytes.toInt(sex));
        System.out.println(Bytes.toInt(age));
        System.out.println(Bytes.toString(address));
    }
}
```

当前的rowkey

```java
System.out.println(result.getRow());
```

设置范围

```java
Scan scan = new Scan();
scan.setStartRow(Bytes.toBytes(1));
scan.setStopRow(Bytes.toBytes("z"));
```

全表扫描加过滤器

列值过滤器（SingleColumnValueFilter，等于，不等于，范围）

```java
@Test
public void scanDataByFilter1() throws Exception {
    //创建过滤器
    SingleColumnValueFilter singleColumnValueFilter = new SingleColumnValueFilter(Bytes.toBytes("info1"), Bytes.toBy

    Scan scan = new Scan();
    //设置过滤器
    scan.setFilter(singleColumnValueFilter);
    ResultScanner scanner = table.getScanner(scan);
    for (Result result : scanner) {
        byte[] value = result.getValue(Bytes.toBytes("info1"), Bytes.toBytes("name"));
        byte[] sex = result.getValue(Bytes.toBytes("info1"), Bytes.toBytes("sex"));
        byte[] address = result.getValue(Bytes.toBytes("info1"), Bytes.toBytes("address"));
        byte[] age = result.getValue(Bytes.toBytes("info1"), Bytes.toBytes("age"));
        System.out.print(Bytes.toString(result.getRow())+";");
        System.out.print(Bytes.toString(value)+";");
        System.out.print(sex+";");
        System.out.print(age+";");
        System.out.print(address+";");
        System.out.println();
    }
}
```

Filter里面四个参数分别穿列族，列，判断条件（等于），值

列名前缀过滤器（ColumnPrefixFilter，过滤指定前缀的列名）

```java
// 创建过滤器
ColumnPrefixFilter columnPrefixFilter = new ColumnPrefixFilter(Bytes.toBytes("name"));
Scan scan = new Scan();
// 设置过滤器
scan.setFilter(columnPrefixFilter);
```

多个列名前缀过滤器（MultipleColumnPrefixFilter）

rowKey过滤器(RowFilter)

```java
@Test
public void scanDataByFilter3() throws Exception {
    Filter f = new RowFilter(CompareFilter.CompareOp.EQUAL, new RegexStringComparator("^wangsenfeng_200")); //匹配以12
}
```

注：列族family，列qualifier,setFilter时候可以传入Filterlist

```java
FilterList filterList = new FilterList(FilterList.Operator.MUST_PASS_ONE);
```

```java
public static enum
  /** !AND */
  MUST_PASS_ALL,
  /** !OR */
  MUST_PASS_ONE
```

必须全过|只要过一个就行