一，查询

1，使用jdbctemplate实现查询操作

```
/*
 * QueryRunner runner = new QueryRuner(datasource);
 * 返回对象
 * runner.query(sql,new BeanHandler<User>(User.class));
 *
 * 返回list集合
 * runner.query(sql,new BeanListHander<User>(User.class))
 *
 * 在dbutils时候，有接口 ResultSetHandler
 * dbutils提供了针对不同的结果实现类
 * */
```

2，jdbctemplate实现查询，有接口rowmapper

jdbctemplate针对这个接口没有提供实现类，得到不同的类型数据需要自己进行数据封装。

一般在一个项目中尽量避免使用多个技术。

3，查询的具体实现

第一个：查询返回某一个值

```
//4,查询操作返回某个操作
        @Test
        public void selectDemo1(){
            //1),创建一个对象，设置登录数据库的相关信息
            DriverManagerDataSource dataSource = new
DriverManagerDataSource();
            dataSource.setDriverClassName("com.mysql.cj.jdbc.Driver");
            dataSource.setUrl("jdbc:mysql://localhost:3306/hb1?
serverTimezone=UTC");
            dataSource.setUsername("root");
            dataSource.setPassword("AQL271422");
            //2),创建一个jdbc模板对象，设置数据源
            JdbcTemplate jdbcTemplate = new JdbcTemplate(dataSource);
            String string = "select count(*) from anqili";
            //调用jdbctemplate中的方法实现
            Integer countInteger = jdbcTemplate.queryForObject(string,
Integer.class);
            System.out.println(countInteger);
        }
```

原始的jdbc代码实现

```
    //4,查询操作返回某个操作
        @Test
        public void selectDemo1(){
            //1),创建一个对象，设置登录数据库的相关信息
            DriverManagerDataSource dataSource = new
DriverManagerDataSource();
            dataSource.setDriverClassName("com.mysql.cj.jdbc.Driver");
```

```
                dataSource.setUrl("jdbc:mysql://localhost:3306/hb1?
serverTimezone=UTC");
                dataSource.setUsername("root");
                dataSource.setPassword("AQL271422");
                //2),创建一个jdbc模板对象，设置数据源
                JdbcTemplate jdbcTemplate = new JdbcTemplate(dataSource);
                String string = "select count(*) from anqili";
                //调用jdbctemplate中的方法实现
                Integer countInteger = jdbcTemplate.queryForObject(string,
Integer.class);
                System.out.println(countInteger);
        }
        //5，jdbc实现连接数据库
        @Test
        public void testJdbc(){
                Connection connection = null;
                PreparedStatement psmtStatement = null;
                ResultSet rSet = null;
                try {
                        //加载驱动
                        Class.forName("com.mysql.cj.jdbc.Driver");
                        //建立连接
                        connection =
DriverManager.getConnection("jdbc:mysql://localhost:3306/hb1?
serverTimezone=UTC", "root","AQL271422");
                        //编写SQL语句
                        String string = "select *from anqili where name=?";
                        //预编译SQL语句
                        psmtStatement = connection.prepareStatement(string);
                        //设置参数值
                        psmtStatement.setString(1,"jane");
                        //执行SQL语句
                        rSet = psmtStatement.executeQuery();
                        //遍历结果集
                        while(rSet.next()){
                                //得到返回结果的值
                                Integer id = rSet.getInt(1);
                                String nameString = rSet.getString(2);
                                System.out.println(id +","+nameString);
                        }

                } catch (Exception e) {
                        // TODO: handle exception
                        e.printStackTrace();
                }
                finally {
                        try {
```

```
                if (rSet!=null) {
                    rSet.close();
                }
            } catch (Exception e2) {
                // TODO: handle exception
                e2.printStackTrace();
            }
            try {
                if (psmtStatement!=null) {
                    psmtStatement.close();
                }
            } catch (Exception e2) {
                // TODO: handle exception
                e2.printStackTrace();
            }
            try {
                if (connection!=null) {
                    connection.close();
                }
            } catch (Exception e2) {
                // TODO: handle exception
                e2.printStackTrace();
            }
        }
```

第二个：查询返回某个对象

测试代码：

```
//5，查询返回某个结果对象
        @Test
        public void testEntity(){
            //1),创建一个对象，设置登录数据库的相关信息
            DriverManagerDataSource dataSource = new
DriverManagerDataSource();
            dataSource.setDriverClassName("com.mysql.cj.jdbc.Driver");
            dataSource.setUrl("jdbc:mysql://localhost:3306/hb1?
serverTimezone=UTC");
            dataSource.setUsername("root");
            dataSource.setPassword("AQL271422");
            //2),创建一个jdbc模板对象，设置数据源
            JdbcTemplate jdbcTemplate = new JdbcTemplate(dataSource);
```

```
                String string = "select *from  anqili where name=?";
                //3，调用jdbctemplate的方法实现
                //第二个参数是接口RowMapper，需要自己写实现接口的类，自己做数据的封装
                user user = jdbcTemplate.queryForObject(string, new RowMappers(),"jack");
        System.out.println(user);
        }
```

自己写的rowmapper类：

```
class RowMappers implements RowMapper<user>{

    @Override
    //rset：查询出来的结果集
    //num：表示查询结果的记录数，表示第几行数据
    public user mapRow(ResultSet rSet, int num) throws SQLException {
        // TODO Auto-generated method stub
        //1，从结果集中拿到数据
        Integer id = rSet.getInt("id");
        String name = rSet.getString("name");
        //2，把得到的数据封装到对象里面
        user user = new user();
        user.setId(id);
        user.setName(name);
        return user;
    }
}
```

第三个：查询返回list集合

测试代码：

```
// 6，查询返回某个结果list
    @Test
    public void testList() {
        // 1),创建一个对象，设置登录数据库的相关信息
        DriverManagerDataSource dataSource = new DriverManagerDataSource();
        dataSource.setDriverClassName("com.mysql.cj.jdbc.Driver");
        dataSource.setUrl("jdbc:mysql://localhost:3306/hb1?serverTimezone=UTC");
        dataSource.setUsername("root");
        dataSource.setPassword("AQL271422");
```

```
        // 2),创建一个jdbc模板对象，设置数据源
        JdbcTemplate jdbcTemplate = new JdbcTemplate(dataSource);
        String string = "select *from  anqili ";
        // 3，调用jdbctemplate的方法实现
        // 查询所有的数据
        List<user> list = jdbcTemplate.query(string, new RowMappers());
        System.out.println(list);
    }
```

自己封装的rowMapper

```
class RowMappers implements RowMapper<user> {

    @Override
    // rset：查询出来的结果集
    // num：表示查询结果的记录数，表示第几行数据
    public user mapRow(ResultSet rSet, int num) throws SQLException {
        // TODO Auto-generated method stub
        // 1，从结果集中拿到数据
        Integer id = rSet.getInt("id");
        String name = rSet.getString("name");
        // 2，把得到的数据封装到对象里面
        user user = new user();
        user.setId(id);
        user.setName(name);
        return user;
    }
}
```