

公告

QQ技术交流群如下, 别在意群名



昵称：张龙豪
园龄：5年3个月
粉丝：3853
关注：21

最新随笔

- 1. 来吧学学.Net Core之登录认证与跨域资源使用
- 2. 来吧学学.Net Core之项目文件简介及配置文件与IOC的使用
- 3. 任正非:我的父亲母亲
- 4. HTTP协议中的短轮询、长轮询、长连接和短连接
- 5. 跨域资源共享CORS详解
- 6. C#各个版本中的新增特性详解
- 7. 仓央嘉措不负如来不负卿
- 8. Redis Sentinel实现的机制与原理详解
- 9. Redis的发布订阅及.NET客户端实现
- 10. .NET客户端实现Redis中的管道(PipeLine)与事物(Transactions)

随笔分类(160)

- .NET Framework/Core (33)
- Asp.Net/Core (21)
- Docker
- ExtJs4 (8)
- Html (3)
- JavaScript (7)
- Linux (9)
- MemCached (1)
- MongoDB (3)
- MSSQL数据库 (18)
- Nginx (4)
- Quartz.Net (3)
- RabbitMQ (2)
- Redis (10)
- TCP/IP协议 (2)
- WCF (3)
- 多线程 (3)

数据库设计三大范式

数据库设计范式

什么是范式：简言之就是，数据库设计对数据的存储性能，还有开发人员对数据的操作都有莫大的关系。所以建立科学的，规范的的数据库是需要满足一些

规范的来优化数据数据存储方式。在关系型数据库中这些规范就可以称为范式。

什么是三大范式：

第一范式：当关系模式R的所有属性都不能在分解为更基本的数据单位时，称R是满足第一范式的，简记为1NF。满足第一范式是关系模式规范化的最低要

求，否则，将有很多基本操作在这样的关系模式中实现不了。

第二范式：如果关系模式R满足第一范式，并且R得所有非主属性都完全依赖于R的每一个候选关键属性，称R满足第二范式，简记为2NF。

第三范式：设R是一个满足第一范式条件的关系模式，X是R的任意属性集，如果X非传递依赖于R的任意一个候选关键字，称R满足第三范式，简记为3NF。

注：关系实质上是一张二维表，其中每一行是一个元组，每一列是一个属性

理解三大范式

第一范式

- 1、每一列属性都是不可再分的属性值，确保每一列的原子性
- 2、两列的属性相近或相似或一样，尽量合并属性一样的列，确保不产生冗余数据。

[workflows \(1\)](#)
 [architecture design \(16\)](#)
 [life cultivation \(2\)](#)
 [design patterns \(11\)](#)
 [data structure](#)

随笔档案(122)

[2017年6月 \(2\)](#)
 [2017年5月 \(3\)](#)
 [2017年4月 \(1\)](#)
 [2017年3月 \(8\)](#)
 [2017年2月 \(8\)](#)
 [2016年7月 \(1\)](#)
 [2016年5月 \(2\)](#)
 [2016年3月 \(4\)](#)
 [2016年2月 \(2\)](#)
 [2016年1月 \(4\)](#)
 [2015年12月 \(4\)](#)
 [2015年11月 \(9\)](#)
 [2015年8月 \(1\)](#)
 [2015年7月 \(3\)](#)
 [2014年12月 \(1\)](#)
 [2014年8月 \(14\)](#)
 [2014年5月 \(3\)](#)
 [2014年4月 \(5\)](#)
 [2014年3月 \(2\)](#)
 [2014年1月 \(1\)](#)
 [2013年8月 \(3\)](#)
 [2013年7月 \(1\)](#)
 [2013年6月 \(1\)](#)
 [2013年5月 \(1\)](#)
 [2013年3月 \(3\)](#)
 [2013年2月 \(1\)](#)
 [2013年1月 \(3\)](#)
 [2012年11月 \(6\)](#)
 [2012年10月 \(1\)](#)
 [2012年8月 \(2\)](#)
 [2012年7月 \(2\)](#)
 [2012年6月 \(1\)](#)
 [2012年5月 \(11\)](#)
 [2012年4月 \(8\)](#)

积分与排名

积分 - 313740
 排名 - 490

最新评论

[1. Re: 聊天程序 \(基于Socket、Thread\)](#)

	Name	Code	Data Type	Length	Precision	P	F	M
1	留言编号	MsgId	int			<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
2	留言人姓名	Name	nvarchar(30)	30		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
3	留言人座机	Telephone	varchar(20)	20		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
4	留言人手机	MobilePhone	varchar(20)	20		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
5	邮箱	Mail	varchar(50)	50		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
6	内容	Content	nvarchar(800)	800		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
7	时间	Time	datetime			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
8	标题	Title	nvarchar(50)	50		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
9	地址	Address	nvarchar(80)	80		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

	Name	Code	Data Type	Length	Precision	P	F	M
	留言编号	MsgId	int			<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
	留言人姓名	Name	nvarchar(30)	30		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	留言人座机	Telephone	varchar(20)	20		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	留言人手机	MobilePhone	varchar(20)	20		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	邮箱	Mail	varchar(50)	50		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	内容	Content	nvarchar(800)	800		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	时间	Time	datetime			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	标题	Title	nvarchar(50)	50		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	省	Province	nvarchar(20)	20		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	市	City	nvarchar(30)	30		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
➔	详细地址	Address	nvarchar(80)	80		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

如果需求知道那个省那个市并按其分类，那么显然第一个表格是不容易满足需求的，也不符合第一范式。

	Name	Code	Data Type	Length	Precision	P	F	M
1	室号	RoomId	int			<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
2	物品1	ProName1	nvarchar(50)	50		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
3	物品1数量	proNum1	int			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
4	物品1价格	ProPrice1	money			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
5	物品2	ProName2	nvarchar(30)	30		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
6	物品2数量	proNum2	int			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
7	物品2价格	ProPrice2	money			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

	Name	Code	Data Type	Length	Precision	P	F	M
1	室号	RoomId	int			<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
2	物品	ProName	nvarchar(50)	50		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
3	物品数量	proNum1	int			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

显然第一个表结构不但不能满足足够多物品的要求，还会在物品少时产生冗余。也是不符合第一范式的。

第二范式

每一行的数据只能与其中一列相关，即一行数据只做一件事。只要数据列中出现数据重复，就要把表拆开来。

看了好几篇龙豪大神写的文章了，我是个新手，不好评价水平，但从讲述逻辑和思路上来看，绝对是一篇好文，另外，单单从注释上面就可以看出文章中满满的诚意，谢谢楼主，博客园第一个评论敬上！

--8月照相馆

2. [Re:SqlServer索引的原理与应用](#)
文章好啊 这评论还挺有意思，一楼一张图片，职业习惯，让我F12看下

--kunge

3. [Re:无刷新分页](#)
[jquery.pagination.js](#)
Jquery插件集

--方缪

4. [Re:asp.net中session的原理及应用](#)
如何向一个服务器端传session?

--绯村剑心、

5. [Re:Jquery表单验证](#)
Jquery插件

--小前端攻城狮

6. [Re:设计模式-抽象工厂模式](#)
工厂类 代码写的不合理 抽象工厂主要是 进行产品线的划分

--烽-火-戏-诸-侯

7. [Re:NLB网路负载均衡管理器详解](#)
博主，我想问您个问题，我在搭建（新建集群，首先你在你局域网中，找一个没有使用过的ip作为你集群的虚拟ip）这步的时候输入IP地址出现不可访问，这该怎么解决？

--爱打野的程序猿

8. [Re:.Net使用Redis详解](#)
[之ServiceStack.Redis（七）](#)
求个app.config中的配置demo

goldthree@qq.com

--GoldThree

9. [Re:.Net使用Redis详解](#)
[之ServiceStack.Redis（七）](#)
求个app.config中的配置demo

--GoldThree

10. [Re:jquery.pagination.js分页](#)
收藏了

--一缕青风`

	Name	Code	Data Type	Length	Precision	P	F	M	^
1	订单编号	= orderId	varchar(30)	30		<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	
2	房间号	roomNum	int			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
3	联系人	name	nvarchar(30)	30		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
4	联系人电话	phone	varchar(15)	15		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
5	身份证号	cardNum	varchar(20)	20		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

一个人同时订几个房间，就会出来一个订单号多条数据，这样子联系人都是重复的，就会造成数据冗余。我们应该把他拆开来。

	Name	Code	Data Type	Length	Precision	P	F	M	^
1	订单编号	= orderId	varchar(30)	30		<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	
2	房间号	roomNum	int			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
3	联系人编号	Peold	id			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

	Name	Code	Data Type	Length	Precision	P	F	M	^
→	联系人编号	Peold	numeric			<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	
2	联系人	name	nvarchar(30)	30		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
3	联系人电话	phone	varchar(15)	15		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
4	身份证号	cardNum	varchar(20)	20		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

这样便实现啦一条数据做一件事，不掺杂复杂的关系逻辑。同时对表数据的更新维护也更易操作。

第三范式

数据不能存在传递关系，即没个属性都跟主键有直接关系而不是间接关系。像：a-->b-->c 属性之间含有这样的关系，是不符合第三范式的。

比如Student表（学号，姓名，年龄，性别，所在院校，院校地址，院校电话）

这样一个表结构，就存在上述关系。 学号--> 所在院校 -->（院校地址，院校电话）

这样的表结构，我们应该拆开来，如下。

（学号，姓名，年龄，性别，所在院校）--（所在院校，院校地址，院校电话）

最后：

三大范式只是一般设计数据库的基本理念，可以建立冗余较小、结构合理的数据库。如果有特殊情况，当然要特殊对待，数据库设计最重

阅读排行榜

1. [存储过程详解\(206791\)](#)

- 2. Lambda表达式详解(183383)
- 3. Nginx配置详解(146167)
- 4. Jquery表单验证(98211)
- 5. Npoi操作excel(64962)
- 6. ExtJs布局详解(57812)
- 7. SQL Server表分区(55590)
- 8. jquery.pagination.js分页(54660)
- 9. ExtJs控件属性配置详细(46711)
- 10. ExtJs4之Grid详细(46577)
- 11. asp.net中session的原理及应用(44025)
- 12. sql语句的优化分析(43950)
- 13. SqlServer索引的原理与应用(43242)
- 14. Linq语法详细(41631)
- 15. c#写windows服务(38266)
- 16. .net反射详解(37484)
- 17. ExtJs4之TreePanel(37045)
- 18. SQL Server中的事务与锁(36461)
- 19. 无刷新分页
jquery.pagination.js(35483)
- 20. 数据库设计三大范式(35046)
- 21. MongoDB高级查询详细(34778)
- 22. 键盘码、ASCII码表(28866)
- 23. SqlServer性能检测和优化工具使用详细(27820)
- 24. asp.net缓存(27775)
- 25. Quartz.net开源作业调度框架使用详解(24731)
- 26. sqlserver 时间格式函数详细(23162)
- 27. IsPostBack原理(21503)
- 28. Html 特殊符号(20045)
- 29. SQL Server游标(20044)
- 30. .Net开源工作流Roadflow的使用与集成(18952)
- 31. Ueditor和CKeditor 两款编辑器的使用与配置(17092)
- 32. Redis数据结构详解之Zset（五）(16265)
- 33. .Net使用Redis详解之ServiceStack.Redis（七）(15733)
- 34. Nginx代理功能与负载均衡详解(14922)
- 35. 聊天程序（基于Socket、Thread）(14705)
- 36. 正则表达式、常用的匹配总结(12412)

要的是看需求跟性能，需求>性能>表结构。所以不能一味的去追求范式建立数据库。

分类: [MSSQL数据库](#)

标签: [数据库三大范式](#)

« 上一篇: [c#写windows服务](#)

» 下一篇: [SqlServer索引的原理与应用](#)

学习这件事
不是缺乏时间
而是缺乏努力

posted @ 2014-04-16 09:02 张龙豪 阅读(35049) 评论(3) 编辑 收藏

评论列表

#1楼 2014-04-16 09:14

、copyman 

如果需求知道那个省那个市并按其分类，那么显然第一个表格是不容易满足需求的，也符合第一范式。
是也不符合吧，要求分解，怎么会符合呢



#2楼[楼主] 2014-04-16

09:50 张龙豪 

@、copyman
是的，笔误，已改，thank。




#3楼 2014-05-11 13:13

唐大侠 

从ORM的角度来看，设计范式和OOP中的独立原则是一样的，我做我的事，别人的事别人做，道理上一样的



[刷新评论](#) [刷新页面](#) [返回顶部](#)

 注册用户登录后才能发表评论，请 [登录](#) 或 [注册](#)，[访问网站首页](#)。

- 37. [HttpHandler与HttpModule的理解与应用\(12052\)](#)
- 38. [ExtJs4 基础必备\(11372\)](#)
- 39. [说说面向服务的体系架构SOA\(10969\)](#)
- 40. [Redis数据结构详解\(一\)\(10301\)](#)

评论**排行榜**

- 1. [浅解.Net分布式锁的实现\(59\)](#)
- 2. [无刷新分页
jquery.pagination.js\(57\)](#)
- 3. [Quartz.net开源作业调度框架使用详解\(46\)](#)
- 4. [.Net中的RealProxy实现AOP\(40\)](#)
- 5. [说说面向服务的体系架构SOA\(39\)](#)
- 6. [存储过程详解\(37\)](#)
- 7. [聊天程序（基于Socket、Thread）\(36\)](#)
- 8. [asp.net缓存\(36\)](#)
- 9. [.Net使用Redis详解
之ServiceStack.Redis（七）\(34\)](#)
- 10. [Lambda表达式详解\(33\)](#)
- 11. [c#操作word文档之简历导出\(32\)](#)
- 12. [.net反射详解\(31\)](#)
- 13. [Nginx配置详解\(29\)](#)
- 14. [.Net开源工作流Roadflow的使用
与集成\(27\)](#)
- 15. [.Net使用RabbitMQ详解\(26\)](#)
- 16. [SQL Server表分区\(26\)](#)
- 17. [Npoi操作excel\(25\)](#)
- 18. [Linq语法详细\(24\)](#)
- 19. [NLB网路负载均衡管理器详解\(24\)](#)
- 20. [asp.net中session的原理及应用\(22\)](#)
- 21. [c#写windows服务\(21\)](#)
- 22. [Quartz.net持久化与集群部署开发详解\(19\)](#)
- 23. [浅解多线程\(18\)](#)
- 24. [缓存依赖（文件、数据库）\(18\)](#)
- 25. [IsPostBack原理\(17\)](#)
- 26. [Application Request Route实现IIS Server Farms集群负载详解\(17\)](#)
- 27. [Jquery表单验证\(16\)](#)
- 28. [SQL Server中的事务与锁\(16\)](#)
- 29. [SqlServer索引的原理与应用\(16\)](#)
- 30. [WCF 应用（一）\(14\)](#)
- 31. [jquery.pagination.js分页\(14\)](#)

- 32. 全国省市区数据库，带拼音, 简称, 行政编码，邮政编码等(13)
- 33. HttpHandler与HttpModule的理解与应用(11)
- 34. ExtJS面向对象(11)
- 35. url重写(11)
- 36. SQL 视图 局部变量 全局变量 条件语句 事务 触发器(10)
- 37. sql语句的优化分析(10)
- 38. SqlServer性能检测和优化工具使用详细(10)
- 39. Redis数据结构详解之List（二）(10)
- 40. memcached安装及.NET中的Memcached.ClientLibrary使用详解(10)

推荐排行榜

- 1. 存储过程详解(158)
- 2. Lambda表达式详解(106)
- 3. .net反射详解(93)
- 4. Quartz.net开源作业调度框架使用详解(79)
- 5. Linq语法详细(63)
- 6. Nginx配置详解(60)
- 7. SQL Server表分区(59)
- 8. SQL Server中的事务与锁(51)
- 9. asp.net缓存(42)
- 10. sql语句的优化分析(39)
- 11. .Net中的RealProxy实现AOP(39)
- 12. SqlServer索引的原理与应用(37)
- 13. JQuery表单验证(36)
- 14. .Net使用RabbitMQ详解(35)
- 15. 聊天程序（基于Socket、Thread）(34)
- 16. HttpHandler与HttpModule的理解与应用(30)
- 17. NLB网路负载均衡管理器详解(30)
- 18. 浅解.Net分布式锁的实现(29)
- 19. 浅解多线程(28)
- 20. 说说面向服务的体系架构SOA(27)
- 21. Npoi操作excel(26)
- 22. Linux系统（一）文件系统、压缩、打包操作总结(26)
- 23. .Net使用Redis详解之ServiceStack.Redis（七）(25)
- 24. memcached安装及.NET中

的Memcached.ClientLibrary使用详解(25)

25. c#写windows服务(24)

26. ExtJs控件属性配置详细(24)

27. ExtJs布局详解(23)

28. c#操作word文档之简历导出(22)

29. 浅解多线程（二）之和尚们的那些事儿(22)

30. .Net开源工作流Roadflow的使用与集成(22)

31. Nginx代理功能与负载均衡详解(20)

32. 数据库设计三大范式(20)

33. C#各个版本中的新增特性详解(20)

34. SqlServer性能检测和优化工具使用详细(19)

35. Application Request Route实现IIS Server Farms集群负载详解(19)

36. Linux系统（三）系统基础扫盲大全(18)

37. SQL Server游标(18)

38. jquery.pagination.js分页(18)

39. 缓存依赖（文件、数据库）(17)

40. asp.net中session的原理及应用(17)