

服务端详细设计

1 总体设计

1.1 概述

服务端提供一个远程调用过程的服务，服务端定义了交互的模式，而客户端使用这些模式，来访问其上的方法，并不需要了解具体的实现上的细节。

1.2 目的

- 通过固定的协议，调用服务端的方法。
- 由于客户端使用 c#语言开发，因此要实现不同程序语言之间的通信，同时也为以后多语言客户端的拓展做准备。

1.3 运行环境

(1)服务器

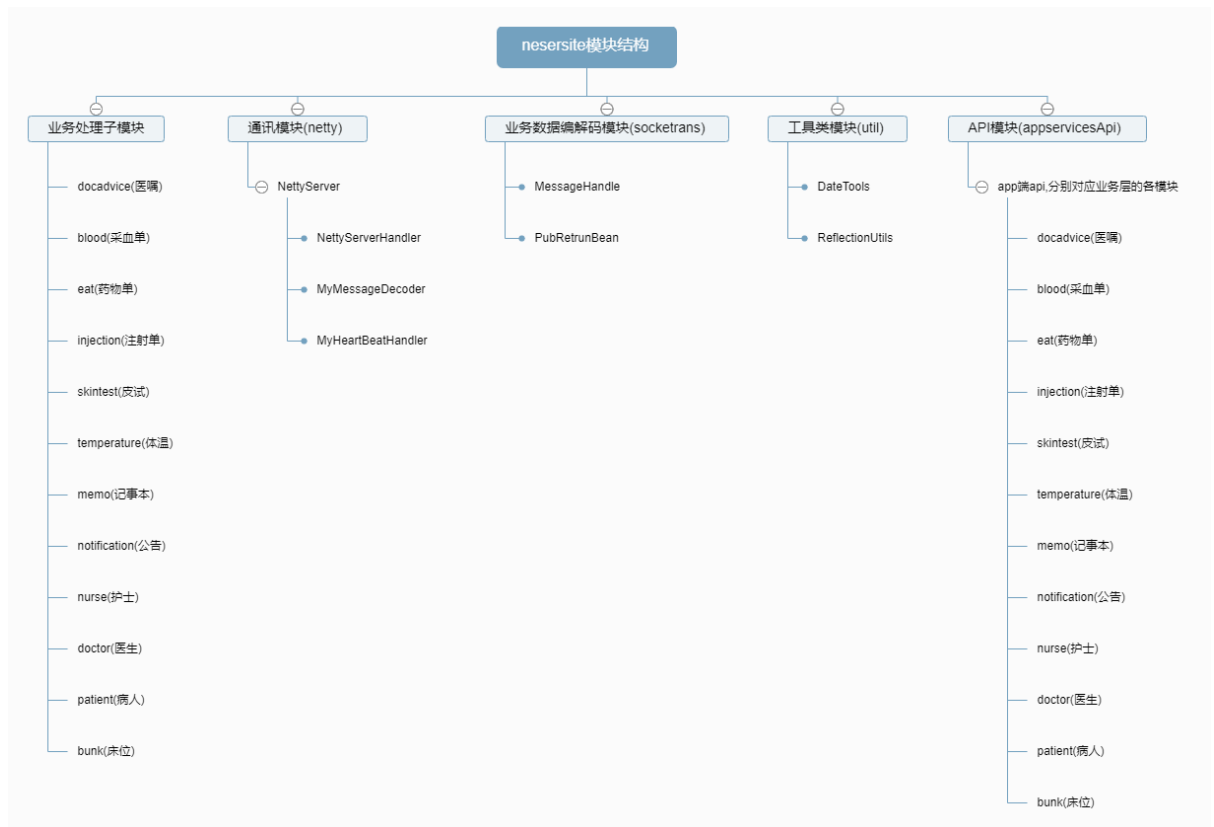
- 操作系统：Windows7 或 WindowsServer2008 及以上版本系统。
- 数据库：Mysql 8.0。
- java 环境：jdk1.8.0_201。

1.4 开发环境

- Eclipse IDE for Java Developers Version: 2019-09 R (4.13.0)

1.5 整体层次架构

如图所示，服务端分为 16 个子模块，包括 12 个供客户端调用的业务处理模块，1 个通讯模块(netty)，1 个业务数据编解码模块(socketrans)，1 个工具类模块(util)，1 个 APP 端的 API 模块。



其中业务层每个子模块又分为三层：分别是 dao 层，service 层，domain 层。
 dao 层对应业务数据相关的数据库操作接口和实现类，
 service 层对应具体业务处理相关的接口和实现类，
 domain 层对应各个业务实体的 Javabean。

1.6 命名规范

- 数据库表名用小写开头的英文命名，字段名用大写开头的英文命名。
- 私有变量以“_”开头的小写英文字母命名，其他变量用小写英文字母开头。
- 所用函数用小写英文字母开头。
- 所有类用大写英文字母开头

1.7 目录规范

形如：**com.naaisi.171206.theNusrseSite.子模块.分层**

com.naaisi.171206.theNusrseSite.Patient.dao.impl
 com.naaisi.171206.theNusrseSite.Patient.service.impl
 com.naaisi.171206.theNusrseSite.Patient. domain

员工 dao 层和实现类
 员工 service 层和实现类
 员工 javabean

1.8 功能模块清单

模块编号	名称	对应目录	模块功能描述
1, 通用模块			
101	通讯模块	netty	调用 netty service 对上层数据进行转发, 自定义协议包, 解决 tcp/ip 沾包半包问题, 提高数据传输稳定性。
102	业务数据编解码模块	socketrans	对业务数据进行 json 编码解码, 编解码所约定的模板详见功能模块设计。
103	工具模块	util/ReflectionUtils	反射工具类, 供业务数据传输解码(socketrans)调用。根据客户端请求的业务数据反向查找对应的业务模块方法, 并返回数据。
2, 业务模块			
201	医嘱模块	docadvice	通过 service 层调取 dao 层的方法对数据进行打包, 包装成适合客户端的医嘱单查询、添加、更改、删除操作, 供客户端调用。
202	输血单模块	blood	通过 service 层调取 dao 层的方法对数据进行打包, 包装成适合客户端的输血单查询、添加、更改、删除操作, 供客户端调用。
203	药物单模块	eat	同上
204	注射单模块	injection	同上
205	皮试单模块	skintest	同上
206	体温模块	temperature	客户端调用, 增加和查询病人相应的体温记录。
207	记事本模块	memo	客户端调用, 增加查询删除护士相应的记事。
208	公告模块	notification	客户端调用, 管理端增加公告并通知各 APP 端。APP 端获取相应的公告记录。
209	护士类模块	nurse	客户端调用, 查询护士相应信息。
210	医生类模块	doctor	客户端调用, 查询医生相应信息。
211	病人类模块	patient	客户端调用, 查询病人相应信息。
212	床位类模块	bunk	客户端调用, 查询病床相应信息。

2 功能模块设计

2.1 概述

各模块的设计说明包括各模块业务流程图、各模块的详细设计说明表、相互关系说明表，从多方面对各个模块的功能、类型、性能、算法逻辑、接口用详细多元精准的表结构进行阐述，以求此种描述方法来让项目文档更加的全面和详细。

2.2 通讯模块

2.2.1 socket 通讯流程图



2.2.2 通讯协议报文格式

MessageProtocol					
4byte	4byte	4byte	8byte	4byte	contentLen
sign	type	status	invoke id	contentLen	content

协议报文消息头详解：

sign：类似 java 字节码文件里的魔数，用来判断是不是 Message 协议的数据包。sign 是常量 2766H（10086D）。

type：标志位，一共 32 个地址位。低四位用来表示消息体数据用的序列化工具的类型（默认为 1、json 业务数据，2、管理端推送消息，0、普通字符串），其余 28 位备用。

status：状态位，设置请求响应状态，定义了一些响应的类型。默认为正常返回 0，异常返回 1。具体类型以后再详细设计。

invoke id：消息 id，long 类型。每一个请求的唯一识别 id（由于采用异步通讯的方式，用来把请求 request 和返回的 response 对应上）

contentLen：消息体 content 长度，int 类型，即记录 Content 有多少个字节

content：消息体的抽象序列化之后存储于此。

2.2.3 模块详细设计说明

模块名称	通讯协议发送模块	模块编号	101.1
模块功能	调用 netty service 对上层数据进行转发		
限制条件	本模块属于通讯协议模块下的一个子模块，执行 message.encoder 方法即执行本模块。		
输入	<ul style="list-style-type: none">待发送的业务数据业务数据对应的编解码器类型状态位，响应类型		
输出	输出项目参考 2.2.2 通讯协议报文格式		
算法逻辑	取得输入项的值并填充到协议报文的对应位置，其中 contentLen 需要通过 Content 计算出来。		
相关对象			

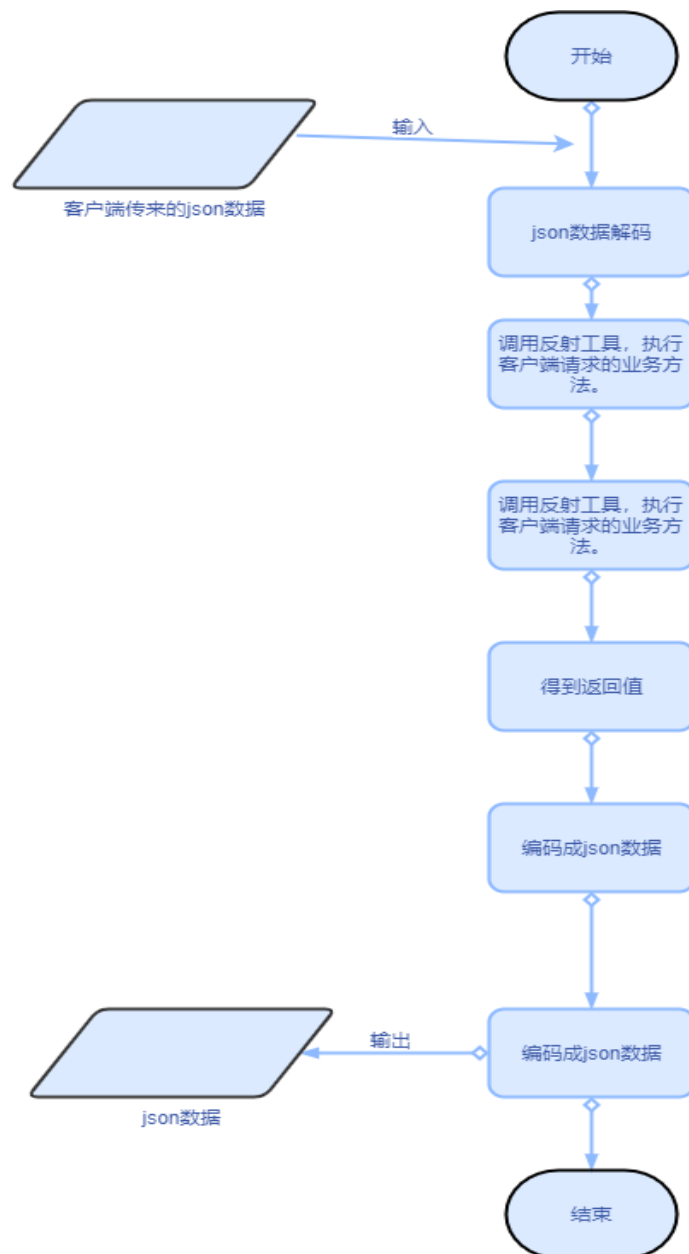
模块名称	通讯协议接收模块	模块编号	101.2
------	----------	------	-------

模块功能	对接收到的通讯报文进行解包。
限制条件	本模块属于通讯协议模块下的一个子模块，执行 message.decoder 方法即执行本模块。
输入	<ul style="list-style-type: none"> ● 参考通讯协议报文
输出	<ul style="list-style-type: none"> ● 接收到的业务数据 ● 业务数据对应的编解码器类型
算法逻辑	首先读取报文 sign 位，判断消息是否属于 message 报文，继续读取 type 位和 contentLen 位，根据 contentLen 读取相应长度的 content，最后将 content 转换成 String。
相关对象	MessageProtocol

模块名称	消息处理模块	模块编号	101.3
模块功能	根据报文类型，对接收到的报文调用相应的解码器。		
限制条件	本模块属于通讯协议模块下的一个子模块，执行 channelRead0 方法即执行本模块。		
输入	<ul style="list-style-type: none"> ● 参考通讯协议报文 		
输出	<ul style="list-style-type: none"> ● 报文消息处理后的反馈结果。 		
算法逻辑	<p>根据报文 type 位，如果报文消息属于业务消息，则开启一下新线程调用 json 业务解码器，并等待解码器返回的结果。</p> <p>如果报文消息属于管理端推送消息则调用推送消息解码器，并等待解码器返回的结果。</p> <p>如果报文消息属于普通字符串消息则调用字符串消息解码器，并等待解码器返回的结果。</p> <p>.....</p>		
相关对象	NettyServerHandler		

2.3 业务数据编解码模块

2.3.1 数据编解码流程图



2.3.2 业务数据(json)字段设计

2.3.2.1接收字段设计,

通过构建 Json 形如:

```
{"remark": "1", "requestClassName": "Patient", "requestID": "001", "requestMethodName": "addPatient", "requestPara": ["testname", "0", "22"]}
```

发起请求。

以下是各个字段的说明.

1. **remark**: 保留字段
2. **requestID**: 请求的 id, 用来区分不同的请求对应的结果。其中 **0-1000 为服务器保留**, 用作服务器主动发起推送时使用的 ID, 如新公告、新医嘱推送到客户端等等。
3. **requestClassName**: 请求的类的名字, 具体的类名字请参考下边的目录。
4. **requestMethodName**: 请求的方法的名字, 每个的类都提供一定的方法供客户端调用, 具体名字和说明请参考 API 说明文档。
5. **requestPara**: 请求的方法的参数, 字符串数组, 如果所调用的方法需要提供参数, 请在此处生成一个 **字符串** 数组, 参数的顺序需要跟提供的方法参数顺序相同。如果没有特别的约定, 所有的填入 json 的参数都默认已经转换成 String。

2.3.2.2发送字段设计

在接收字段的基础上, 增加了 result 字段。

result 字段内包含请求的一个对象或者对象列表。形如:

[对象 1 名称 [对象属性名 1=属性值 1, 对象属性名 2=属性值 2.....], 对象 2 名称 [对象属性名 1=属性值 1, 对象属性名 2=属性值 2.....].....]

具体对象的名称和属性请参考 API 说明。

2.3.3 模块详细设计说明

模块名称	业务数据编码模块	模块编号	102.1
模块功能	对业务数据进行 json 编码		
限制条件	本模块属于业务数据编解码模块下的一个子模块，执行 <code>jsonCodec.jsonEncode</code> 方法即执行本模块。		
输入	<ul style="list-style-type: none">● requestID● requestClassName● requestMethodName● remark● requestPara● result (输入字段说明请参考 2.3.2.2 发送字段设计)		
输出	(输出请参考 2.3.2.1 接收字段设计)		
算法逻辑	根据填入字段生成相关的 bean,调用 fastjosn 等的第三方库，将 bean 转换成 json。		
相关对象	Jsonbean		

模块名称	业务数据解码模块	模块编号	102.2
模块功能	对业务数据进行 json 编码		
限制条件	本模块属于业务数据编解码模块下的一个子模块，执行 <code>jsonCodec.jsonDecode</code> 方法即执行本模块。		
输入	请求的 json 业务数据 (输入字段说明请参考 2.3.2.2 接收字段设计)		
输出	<pre>Jsonbean{ requestID;// 随机数, requestClassName;// 请求的类名 requestMethodName;// 请求的方法名 requestPara;// 请求参数集 字符串数组 }</pre>		
算法逻辑	从请求的 json 字符串获取相关字段并填入 bean,其中 requestPara 的生成，需要根据 requestClassName、requestMethodName 反射找到对应方法的参数类型列表，并将传入参数转换成对应类型，返回一个参数 Object 数组。		
相关对象	Jsonbean		

2.4 工具模块

2.4.1 模块详细设计说明

模块名称	反射类执行模块	模块编号	103.1
模块功能	查找业务相应的方法并执行。		
限制条件	本模块属于工具模块下的一个子模块, 执行 executeRefle 方法即执行本模块。		
输入	<ul style="list-style-type: none">● classname 所需执行的方法所在的类路径● MethodName 所需执行的方法名● args 所需执行的方法参数数组		
输出	方法的返回结果		
算法逻辑	通过反射获得相应的类的方法, 判断传递的参数个数是否与方法参数的个数相同, 可能存在重载的方法。如果没有重载的方法, 则执行, 并返回结果。		
相关对象	ReflectionUtils		

2.5 业务模块

- docadvice(医嘱)
- blood(采血单)
- eat(药物单)
- injection(注射单)
- skintest(皮试)
- temperature(体温)
- memo(记事本)
- notification(公告)
- nurse(护士)
- doctor(医生)
- patient(病人)
- bunk(床位)
- password(密码)

业务模块主要是根据 1.5 中整体层次架构的业务子模块, 建立相应的 domain 实体类, service 服务类, 和 dao 数据库操作类。

dao 数据层主要是采用 hibernate 框架生成对应的增删改查方法。

service 服务类主要是根据需求分析中的用例提取出来的方法, 供客户端调用。因篇幅过大, 此处不再详述, 具体设计说明请参考附件: API 说明。此处只列出方法的目录。

服务端 API 说明.....	1
requestClassName: Blood.....	4
* 添加存储一个 blood 记录.....	4
* 通过采血单 ID 将状态标记为已执行,并记录当前系统时间为执行时间.....	4
* 获取所有采血单.....	4
* 获取所有未执行的采血单.....	4
* 通过 bloodID 获取一个 blood 记录.....	4
* 通过 bloodID 获取该病人所有的 blood 记录.....	5
* 获取该病人所有未执行的 blood 记录.....	5
requestClassName: Bunk.....	5
* // 新增一张床位.....	5
* // 绑定床位给病人。.....	5
* // 获取所有床位.....	5
* // 查找所有空闲的床位.....	6
* //通过 ID 获取一个床位记录.....	6
* //获取一张空闲的床位记录.....	6
* //通过该病人所有的床位记录.....	6
* //将床位设定为已使用.....	6
* //将床位设定为未使用.....	6
requestClassName: DocAdvice.....	7
* 添加一张医嘱 （管理端使用）.....	7
* 通过 ID 将状态标记为已执行,记录执行护士，并记录当前系统时间为执行时间.....	7
* 获取所有的医嘱.....	7
* 获取所有未执行的医嘱.....	7
* 获取一张医嘱.....	8
* 获取病人的所有医嘱.....	8
* 获取病人的所有未执行医嘱.....	8
* 获取护士的所有医嘱.....	8
* 查找护士的所有未执行医嘱.....	8
* 删除改病人所有医嘱 （管理端使用）.....	8
* 删除一张医嘱 （管理端使用）.....	9
requestClassName: Doctor.....	9
* 获取所有医生.....	9
* 获取一个医生.....	9
* 通过姓名获取医生列表.....	9
requestClassName: Eat.....	9
/**添加存储一个口服药物记录.....	9
* 通过口服药物 ID 将状态标记为已执行,并记录当前系统时间为执行时间.....	10
* 获取所有口服药物单.....	10
* 获取所有未执行的 Eat 单.....	10
* 通过 EatID 获取一个 Eat 记录.....	10
* 通过 EatID 获取该病人所有的 Eat 记录.....	10
* 获取该病人所有未执行的 Eat 记录.....	10
requestClassName: Injection.....	11
* 添加存储一个注射药物记录.....	11
* 通过注射药物 ID 将状态标记为已执行,并记录当前系统时间为执行时间.....	11
* 获取所有注射药物单.....	11
* 获取所有未执行的注射药物单.....	11
* 通过 EatID 获取一个注射药物记录.....	11

* 添加存储一个注射药物记录.....	11
* 通过注射药物 ID 将状态标记为已执行,并记录当前系统时间为执行时间.....	11
* 获取所有注射药物单.....	11
* 获取所有未执行的注射药物单.....	11
* 通过 EatID 获取一个注射药物记录.....	11
* 通过 EatID 获取该病人所有的注射药物记录.....	12
* 获取该病人所有未执行的注射药物记录.....	12
requestClassName: Memo.....	12
/**添加备忘记录 权限管理未完成,默认添加到 07 用户的备忘.....	12
* 将备忘状态修改为已完成。1 已完成, 0 未完成.....	12
* 将备忘状态修改为未完成。1 已完成, 0 未完成.....	12
* 获取用户所有备忘记录, 权限管理未完成,默认返回 07 用户的备忘.....	12
requestClassName: Notification.....	13
* 添加一条公告 (管理端使用).....	13
* 推送公告 (管理端使用).....	13
* 获取所有公告.....	13
* 获取最新 5 条公告 (客户端使用).....	13
* 获取一条公告 (管理端使用).....	13
* 删除一条公告 (管理端使用).....	13
requestClassName: Nurse.....	14
* 获取所有护士.....	14
* 获取一个护士.....	14
* 通过姓名获取护士列表.....	14
requestClassName: Patient.....	14
/**添加一个病人.....	14
/**通过 patient id 更新一个病人的信息, 姓名, 性别, 年龄.....	14
/**通过 patient id 更新一个病人的信息, 床位.....	15
/**获取所有病人.....	15
/**获取一个病人.....	15
/**通过姓名获取病人列表.....	15
/**删除一个病人.....	15
requestClassName: SkinTest.....	15
* 添加存储一个皮试记录.....	15
* 通过皮试 ID 将状态标记为已执行,并记录当前系统时间为执行时间.....	15
* 获取所有皮试单.....	16
* 获取所有未执行的 SkinTest 单.....	16
* 通过 SkinTestID 获取一个 SkinTest 记录.....	16
* 通过 SkinTestID 获取该病人所有的 SkinTest 记录.....	16
* 获取该病人所有未执行的 SkinTest 记录.....	16
requestClassName: Temperature.....	17
* 将体温记录修改为无效。1 有效, 0 无效.....	17
* 将体温记录修改为有效。1 有效, 0 无效.....	17
* 获取病人所有体温记录.....	17
* 获取病人最近 15 条体温记录.....	17

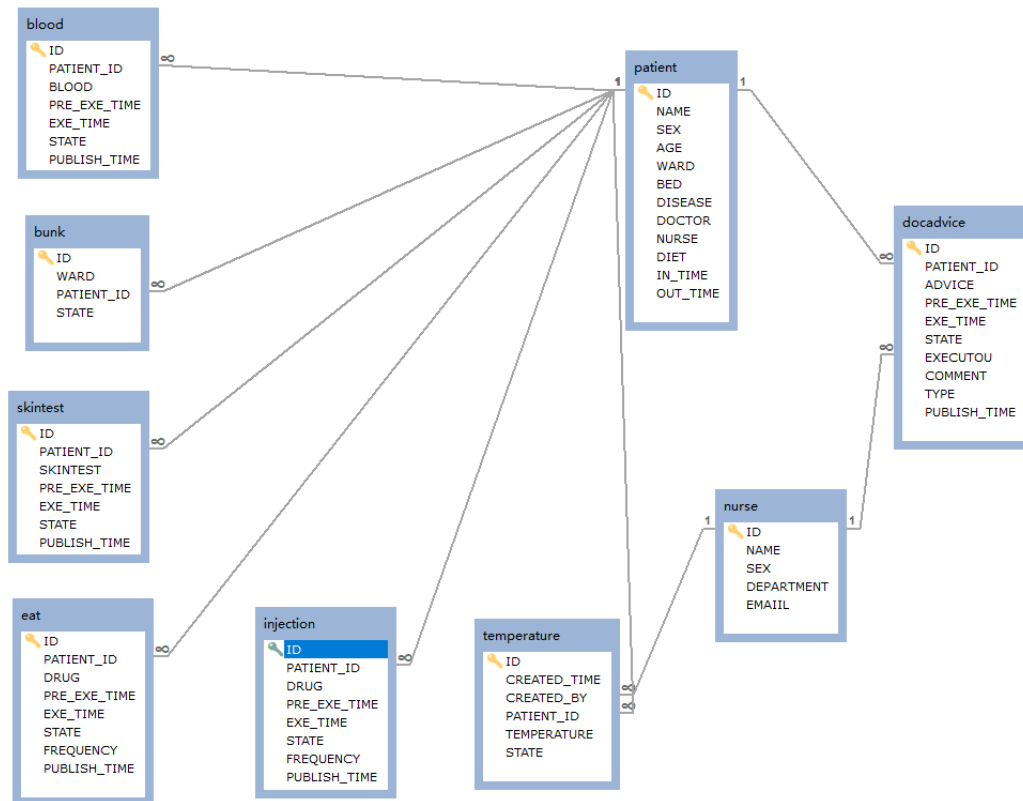
3 数据库设计

3.1 数据库中表名列表

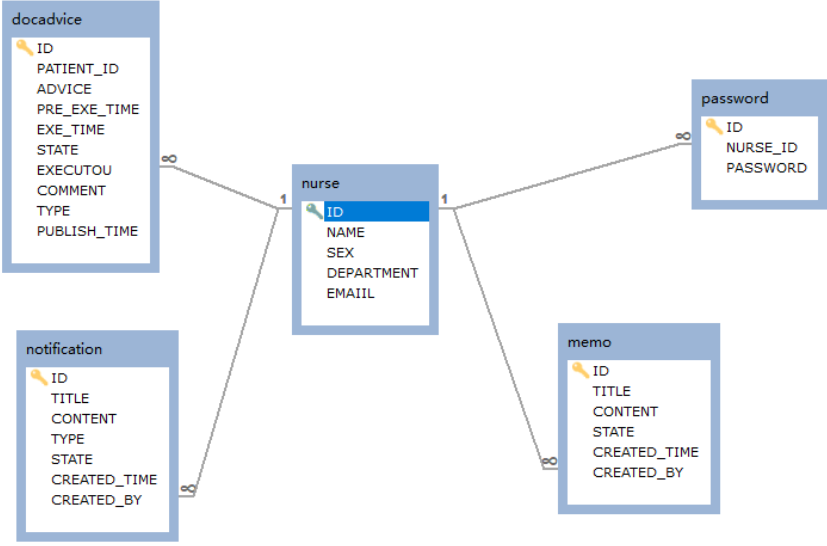
- docadvice(医嘱)
- blood(采血单)
- eat(药物单)
- injection(注射单)
- skintest(皮试)
- temperature(体温)
- memo(记事本)
- notification(公告)
- nurse(护士)
- doctor(医生)
- patient(病人)
- bunk(床位)
- password(密码)

3.2 数据库表之间的关系

3.2.1 patient 表与其他表的关系：



3.2.2 nurse 表与其他表的关系:




3.3 数据库表的详细清单


- docadvice(医嘱)

	Field	Type	Comment
🔑	ID	int NOT NULL	任务ID
	PATIENT_ID	int NOT NULL	病人ID
	ADVICE	varchar(3072) NOT NULL	医嘱内容
	PRE_EXE_TIME	datetime NULL	预计执行时间
	EXE_TIME	datetime NULL	实际执行时间
	STATE	int NULL	完成状态 0未完成, 1已执行, 2归档
	EXECUTOU	int NULL	执行护士
	COMMENT	varchar(3072) NULL	备注
	TYPE	int NULL	医嘱类型 0临时, 1长期
	PUBLISH_TIME	datetime NULL	发布时间


- blood(采血单)

	Field	Type	Comment
	ID	int NOT NULL	任务ID
	PATIENT_ID	int NOT NULL	病人ID
	BLOOD	varchar(128) NOT NULL	采血项目
	PRE_EXE_TIME	datetime NULL	预计执行时间
	EXE_TIME	datetime NULL	实际执行时间
	STATE	int NULL	完成状态
	PUBLISH_TIME	datetime NULL	发布时间


- eat(药物单)

	Field	Type	Comment
	ID	int NOT NULL	任务ID
	PATIENT_ID	int NOT NULL	病人ID
	DRUG	varchar(32) NOT NULL	药物名称
	PRE_EXE_TIME	datetime NULL	预计执行时间
	EXE_TIME	datetime NULL	实际执行时间
	STATE	int NULL	完成状态
	FREQUENCY	varchar(32) NULL	频次
	PUBLISH_TIME	datetime NULL	发布时间


- injection(注射单)

	Field	Type	Comment
	ID	int NOT NULL	任务ID
	PATIENT_ID	int NOT NULL	病人ID
	DRUG	varchar(128) NOT NULL	药物名称
	PRE_EXE_TIME	datetime NULL	预计执行时间
	EXE_TIME	datetime NULL	实际执行时间
	STATE	int NULL	完成状态
	FREQUENCY	varchar(32) NULL	频次
	PUBLISH_TIME	datetime NULL	发布时间


- skintest(皮试)

	Field	Type	Comment
	ID	int NOT NULL	任务ID
	PATIENT_ID	int NOT NULL	病人ID
	SKINTEST	varchar(128) NOT NULL	皮试项目
	PRE_EXE_TIME	datetime NULL	预计执行时间
	EXE_TIME	datetime NULL	实际执行时间
	STATE	int NULL	完成状态
	PUBLISH_TIME	datetime NULL	发布时间


- temperature(体温)

	Field	Type	Comment
	ID	int NOT NULL	条目ID
	CREATED_TIME	datetime NOT NULL	创建时间
	CREATED_BY	int NULL	创建人
	PATIENT_ID	int NOT NULL	绑定病人ID
	TEMPERATURE	decimal(32,10) NOT NULL	温度记录
	STATE	int NULL	状态


- memo(记事本)

	Field	Type	Comment
	ID	int NOT NULL	备忘条目ID
	TITLE	varchar(128) NULL	标题
	CONTENT	varchar(3072) NULL	内容
	STATE	int NULL	状态
	CREATED_TIME	datetime NULL	创建时间
	CREATED_BY	int NOT NULL	创建人


- notification(公告)

	Field	Type	Comment
	ID	int NOT NULL	通知ID
	TITLE	varchar(128) NOT NULL	标题
	CONTENT	varchar(3072) NULL	内容
	TYPE	int NULL	分类
	STATE	int NOT NULL	状态
	CREATED_TIME	datetime NULL	创建时间
	CREATED_BY	int NOT NULL	创建人


- nurse(护士)

	Field	Type	Comment
	ID	int NOT NULL	工号ID
	NAME	varchar(128) NOT NULL	姓名
	SEX	int NULL	性别
	DEPARTMENT	varchar(128) NULL	所属部门
	EMAIL	varchar(32) NULL	邮箱

- doctor(医生)

	Field	Type	Comment
	ID	int NOT NULL	工号ID
	NAME	varchar(128) NOT NULL	姓名
	SEX	int NULL	性别
	DEPARTMENT	varchar(32) NULL	所属部门
	EMAIL	varchar(128) NULL	邮箱


- patient(病人)

	Field	Type	Comment
	ID	int NOT NULL	Id
	NAME	varchar(128) NOT NULL	姓名
	SEX	int NULL	性别
	AGE	int NULL	年龄 0男1女
	WARD	varchar(128) NULL	病区
	BED	varchar(32) NULL	床号
	DISEASE	varchar(128) NULL	病名
	DOCTOR	int NULL	主治医师ID
	NURSE	int NULL	负责护士ID
	DIET	varchar(512) NULL	饮食要求
	IN_TIME	datetime NULL	入院时间
	OUT_TIME	datetime NULL	出院时间

- bunk(床位)

	Field	Type	Comment
	ID	int NOT NULL	床位ID
	WARD	varchar(32) NULL	所属病区
	PATIENT_ID	int NULL	绑定病人ID
	STATE	int NOT NULL	状态 0空闲，1已使用，2其他

- password(密码)

	Field	Type	Comment
	ID	int NOT NULL	
	NURSE_ID	int NOT NULL	
	PASSWORD	varchar(1024) NOT NULL	