# 数据库详细设计

## 数据库表项设计

根据对系统的分析。数据库总体来可设计成如下几个表项。

(1) 学校节点表(USERINFO)

该表主要用于存储学校节点信息，各个节点用于配置网络测量项目所需的服务；表中属性主要包括节点名和节点IP，具体如表1-1所示：

表1-1 学校节点表(networkmeasurement\_schoolnode)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 是否可空 | 字段含义 | 特殊说明 |
| id | INT(11) | Not Null | 表项id | 主键，自增，唯一区分每条记录 |
| nodeName | VARCHAR(50) | Not Null | 节点名 | 用于记录部署节点的名称（如，上海交通大学，复旦大学） |
| nodeIp | CHAR(15) | Not Null | 节点IP | 用于记录部署节点的IP |

(2) 协议表(networkmeasurement\_netprotocol)

该表主要用于记录主动测量中需要用到的TCP/IP协议，具体包括TCP,UDP,ICMP,SNMP(被动测量中默认使用该协议)，具体如表2-2所示。

表3-2 订单表(networkmeasurement\_netprotocol)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 是否可空 | 字段含义 | 特殊说明 |
| id | INT(11) | Not Null | 表项id | 主键，自增，唯一区分每条记录 |
| protocolName | VARCHAR(10) | Not Null | 协议名称 | 用于记录项目中所用到的TCP/IP协议（UDP，TCP，ICMP等） |

(3) 主动测量表(networkmeasurement\_active)

该表主要用于记录主动测量的网络指标；

带宽：本地测试服务器至远端测试服务器当前网络可达的最大发送带宽；

延迟：发送时延(包大小/发送速度)+排队时延+传播时延（长度/速度）；

抖动：用来描述端到端的数据包分组的延迟变化程度;Jitter=( | packet\_n - packet\_n-1 |+| packet\_n-1 - packet\_n-2 | + ...+ | packet\_2 - packet\_1|)/n ；

丢包率：成功发送数据包数/发送数据包总数；

可用性：当前网络是否可用：连通、可达。

具体数据库表如2-3所示。

表2-3 主动测量表(networkmeasurement\_active)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 是否可空 | 字段含义 | 特殊说明 |
| id | INT(11) | Not Null | 主动测量记录号 | 主键，自增，唯一区分每条记录 |
| startNode\_id | INT(11) | Not Null | 起始节点id | 外键，关联表项networkmeasurement\_schoolnode |
| endNode\_id | INT(11) | Not Null | 终止节点id | 外键，关联表项networkmeasurement\_schoolnode |
| protocol\_id | INT(11) | Not Null | 协议id | 外键，关联表项networkmeasurement\_netprotocol |
| createTime | DATETIME | Not Null | 创建记录时间 | 记录该次主动测量的时间戳 |
| bandwidth | DOUBLE | Not Null | 带宽 | 最大发送带宽(单位：Mps) |
| delay | DOUBLE | Not Null | 单向时延 | 起始节点到终止节点的单向时延(单位：ms) |
| jitter | DOUBLE | Not Null | 抖动 | 记录网络时延变化的程度(单位：ms) |
| loss | DOUBLE | Not Null | 丢包率 | 记录当前网络的丢包情况(单位：%) |
| congestion | TINYINT(1) | Not Null | 拥塞 | 描述当前网络是否拥塞，bool类型，这里认为超过%0.1认为当前网络拥塞 |
| avail | TINYINT(1) | Not Null | 可用性 | 描述当前网络是否可用，连通，类型：bool； |

(4) 被动测量表(networkmeasurement\_passive)

该表主要用于记录被动测量（利用SNMP协议）时获得的网络性能指标；

带宽：记录起始节点到终止节点该条链路中经过的路由器的最小带宽；

吞吐量：记录起始节点到终止节点该条链路中经过的路由器的最大吞吐量；

延迟：记录起始节点到终止节点该条链路的往返时延；

丢包率：成功发送数据包数/发送数据包总数；

Cpu利用率：记录起始节点到终止节点该条链路中经过的路由器的最大cpu利用率；

内存利用率：记录起始节点到终止节点该条链路中经过的路由器的最大内存利用率；

具体数据库表2-4所示。

表3-4被动测量表(networkmeasurement\_passive)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 是否可空 | 字段含义 | 特殊说明 |
| id | INT(11) | Not Null | 主动测量记录号 | 主键，自增，唯一区分每条记录 |
| startNode\_id | INT(11) | Not Null | 起始节点id | 外键，关联表项networkmeasurement\_schoolnode |
| endNode\_id | INT(11) | Not Null | 终止节点id | 外键，关联表项networkmeasurement\_schoolnode |
| createTime | DATETIME | Not Null | 创建记录时间 | 记录该次主动测量的时间戳 |
| bandwidth | DOUBLE | Not Null | 带宽 | 记录起始节点到终止节点该链路的物理带宽(单位：Mps) |
| throughput | DOUBLE | Not Null | 吞吐量 | 起始节点到终止节点该链路在一定时间内通过的数据包(单位：Bps) |
| loss | DOUBLE | Not Null | 丢包率 | 记录起始节点到终止节点该链路的丢包情况(单位：%) |
| rtt | DOUBLE | Not Null | 往返时延 | 记录起始节点到终止节点该链路的往返时延(单位：ms) |
| cpu | DOUBLE | Not Null | cpu利用率 | 记录起始节点到终止节点该链路经过路由的cpu利用率(单位：%) |
| memory | DOUBLE | Not Null | 内存利用率 | 记录起始节点到终止节点该链路经过路由的内存利用率(单位：%) |
| ip\_bandwidth | CHAR(15) | Not Null | 瓶颈带宽地址 | 记录起始节点到终止节点该链路最小带宽的路由地址 |
| ip\_throughput | CHAR(15) | Not Null | 瓶颈吞吐量地址 | 记录起始节点到终止节点该链路最大吞吐量的路由地址 |
| ip\_cpu | CHAR(15) | Not Null | 瓶颈带宽地址 | 记录起始节点到终止节点该链路最大cpu利用率的路由地址 |
| ip\_memory | CHAR(15) | Not Null | 瓶颈带宽地址 | 记录起始节点到终止节点该链路最大内存利用率的路由地址 |

(5) 系统管理员表(ADMININFO)

用于系统管理员登录，对系统进行维护和管理，仅需账号和密码即可，如表2-5所示。

## 数据库关系图

数据库各个表项的关系如图表关系如图1-1所示。

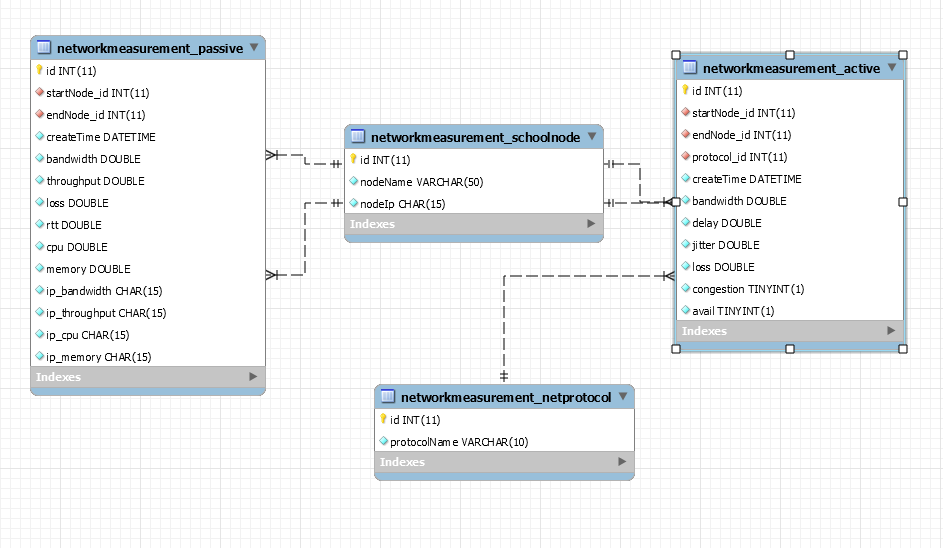


图1-1 数据库关系图

其中，主键外键的对应关系可如下表1-5所示：

|  |  |  |  |
| --- | --- | --- | --- |
| 主键 | 外键 | 主键所在表 | 外键所在表 |
| startNode\_id | id | networkmeasurement\_active | networkmeasurement\_schoolnode |
| endNode\_id | id | networkmeasurement\_active | networkmeasurement\_schoolnode |
| protocol\_id | id | networkmeasurement\_active | networkmeasurement\_netprotocol |
| startNode\_id | id | networkmeasurement\_passive | networkmeasurement\_schoolnode |
| endNode\_id | id | networkmeasurement\_passive | networkmeasurement\_schoolnode |

## SQL语言

1. 创建database:

create database network;

1. 创建各个表项：

这里利用Django自带的models模块进行表项的创建，而非手动用sql语言；

1).在Django项目的settings.py中配置数据库：

DATABASES = {

'default': {

'ENGINE': 'django.db.backends.mysql',

'NAME': 'network',

'USER': 'root',

'PASSWORD':'root',

'HOST':'127.0.0.1',

'PORT':3306,

}

}

2).在models.py中配置表项：

#school node:

class SchoolNode(models.Model):

nodeName = models.CharField(max\_length=50)

#nodeIp = models.CharField(max\_length=16)

nodeIp = models.IPAddressField() #alse char(15)

def \_\_unicode\_\_(self):

return self.nodeName

#protocol:TCP,UDP,ICMP,SNMP

class NetProtocol(models.Model):

protocolName = models.CharField(max\_length=10)

def \_\_unicode\_\_(self):

return self.protocolName

#to store active\_networkMeasurement informations

class Active(models.Model):

startNode = models.ForeignKey(SchoolNode,related\_name="start\_node")

endNode = models.ForeignKey(SchoolNode,related\_name="end\_node")

protocol = models.ForeignKey(NetProtocol)

createTime = models.DateTimeField()

bandwidth = models.FloatField(default=0)

delay = models.FloatField(default=0)

jitter = models.FloatField(default=0)

loss = models.FloatField(default=0)

congestion = models.BooleanField(default=False)

avail = models.BooleanField(default=False)

#passive recodes

class Passive(models.Model):

startNode = models.ForeignKey(SchoolNode,related\_name="passive\_start\_node")

endNode = models.ForeignKey(SchoolNode,related\_name="passive\_end\_node")

createTime = models.DateTimeField()

bandwidth = models.FloatField(default=0)

throughput = models.FloatField(default=0)

loss = models.FloatField(default=0)

rtt = models.FloatField(default=0)

cpu = models.FloatField(default=0)

memory = models.FloatField(default=0)

ip\_bandwidth = models.IPAddressField(default="127.0.0.1")

ip\_throughput = models.IPAddressField(default="127.0.0.1")

ip\_cpu = models.IPAddressField(default="127.0.0.1")

ip\_memory = models.IPAddressField(default="127.0.0.1")