**北京邮电大学课程设计报告**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **课程设计**  **名称** | **数据库系统原理课程设计** | | **学 院** | **计算机学院** | **指导教师** | **叶文** |
| **班 级** | **班内序号** | **学 号** | | **学生姓名** | **成绩** | |
| **2015211312** | **10** | **2015211471** | | **祁瑞** |  | |
| **2015211312** | **4** | **2015211465** | | **张嘉轩** |  | |
| **课**  **程**  **设**  **计**  **内**  **容** | 教学目的：本次课程设计综合考察了上学期所学的内容，包括数据库系统，软件工程，现代交换原理等课程。通过课程设计，有机会将学到的知识在实际分析中运用，将理论与实践相结合。  基本内容：开发数据库应用系统实现以下功能：1实现用户注册功能；2数据导入导出功能  3 信息查询；4 信息分析  实验方法：利用C++语言开发sql server数据库应用系统  团队分工：祁瑞：图形化界面的设计，绘制图表，数据导入导出，信息查询  张嘉轩：用户注册登录，存储过程，干扰分析 | | | | | |
| **学生**  **课程设计**  **报告**  （附页） |  | | | | | |
| **课**  **程**  **设**  **计**  **成**  **绩**  **评**  **定** | 遵照实践教学大纲并根据以下四方面综合评定成绩：   1. 课程设计目的任务明确，选题符合教学要求，份量及难易程度   2、团队分工是否恰当与合理  3、综合运用所学知识，提高分析问题、解决问题及实践动手能力的效果  4、是否认真、独立完成属于自己的课程设计内容，课程设计报告是否思路清晰、文字通顺、书写规范  **评语**:    **成绩**:  指导教师签名：  年 月 日 | | | | | |

注：评语要体现每个学生的工作情况，可以加页。

# 一、实验目的

1. 进一步掌握数据库应用系统DBAS的设计过程和生命周期

2. 掌握以应用程序的方式实现数据的规格化导入和导出

3. 学会优化sql查询处理语句与优化

4. 掌握存储过程和函数在数据库应用系统中的应用

# 二、实验内容

## 1．系统功能需求分析

### 用户管理

该应用系统支持新用户注册，用户可根据用户名和密码登陆系统。

### 1.2 数据导入功能

编写数据导入程序，导入网络配置信息、KPI指标信息、PRB干扰信息、MRO数据4种数据。导入功能实现excel和csv两种格式的文件导入。

导入数据时，要求数据库中已有对应的表结构。向数据库表导入数据时，根据导入的数据在数据库中是否已经存在，采取insert或update动作。导入过程中必须采用批量导入及导入触发器等技术，以实现数据快速导入。需要开发导入程序和定义在数据库表上的导入触发器，2者共同完成数据导入工作。

### 1.3 数据导出功能

在界面上用一个下拉列表显示数据库中的表，指定某些表，如tbCell，将

表中数据以Excel或txt格式的文件输出，要求输出文件中应当有表中各个属性名称。数据导出时，可以自定义导出文件名，选择文件存放路径。

### 1.4 信息查询功能

（1）小区配置信息查询

从tbCell表中查询小区信息。用户在界面上以输入框或者下拉列表的方式，选择小区ID Cell\_ID/Sector\_ID或小区名称CellName/SectorName，以列表方式输出该小区全部信息；程序在查询界面下拉列表中列出表tbCell中全部小区名称，用户在界面上输入框或者下拉列表的方式，以列表方式输出该小区全部信息。

（2）基站eNodeB信息查询

从tbCell表中查询基站所属全部小区信息。用户在界面上以输入框或者下拉列表的方式选择基站ID或基站名称，以列表方式输出该基站全部小区信息。

（3）KPI指标信息查询

用户在界面上根据网元名称以输入框或者下拉列表的方式选定表tbKPI中的某个特定网元，在右侧显示其属性列表，通过时间/日历控件，查询网元某个时间段（天级）某个属性值的变化情况，并将结果用柱状图或折线图的方式呈现出来，例如查询网元从2016年7月17日到7月19日的RRC连接重建比率 (%)。

（4）PRB信息统计与查询

根据表“优化区17日-19日每PRB干扰 查询-15分钟”，统计小时级的PRB干扰数据，生成一张新表tbPRBnew，并导出到外部的excel表中。

根据得到的小时级的PRB干扰数据表tbPRBnew，在界面上根据网元名称以输入框或者下拉列表的方式选定某个特定网元，在右侧显示其属性列表，添加时间控件，查询网元某个时间段（小时级）某个属性值的变化情况，并将结果用柱状图或折线图的方式呈现出来，例如查询网元某一天从01:00:00到13：00：00每个小时内第60个PRB上检测到的干扰噪声的平均值。也可将时间段缩小到分钟级，此时用tbPRB进行查询，总之查询的时间粒度要和表对应。

所有查询内容都要求可以导出，查询结果的图表要求可以导出，保存到可选路径中。

### 1.5 干扰分析

具体要求和实现步骤如下：

（1）主邻小区RSRP差值（即C2I）的均值C2I\_mean、标准差std计算

根据导入的MRO测量报告数据表tbMROData表中主小区和邻小区的参考信号接收功率RSRP，计算主邻小区RSRP差值，该联系是多对多联系。计算每对主邻小区<SCell, NCell>的RSRP差值C2I，并计算出C2I的均值和标准差。

（2）主邻小区RSRP差值小于9的概率PrbC2I9计算

根据每对主邻小区RSRP差值的均值和标准差，可以得到正态分布，根据正态分布求出主邻小区RSRP差值小于9的概率Prb9=Pr{C2I<9db}。

（3）主邻小区RSRP差值绝对值小于6的概率PrbABS6的计算

根据每对主邻小区RSRP差值的均值和标准差，可以得到正态分布，根据正态分布求出主邻小区RSRP差值绝对值小于6的概率（PrbABS6）。

（4）生成新表tbC2Inew

新表包括六个属性：主小区ID、邻小区ID、主邻小区RSRP差值的均值、主邻小区RSRP差值的标准差、主邻小区RSRP差值小于9的概率、主邻小区RSRP差值绝对值小于6的概率。

注：可自己设置一个参数，将<LteScRSRP, LteNcRSRP>RSRP测量值对条数小于这个参数的内容先筛选掉。

### 1.6 查询重叠覆盖干扰三元组

根据表tbC2Inew，找出所有的小区三元组<a,b,c>（其中a,b,c互为邻小区），生成新表tbC2I3，其中有三个属性，分别是三个小区的小区ID。

使得：

1. PrbABS6(a,b)或PrbABS6(b,a)>=x%;
2. PrbABS6(a,c)或PrbABS6(c,a)>=x%;
3. PrbABS6(b,c)或PrbABS6(c,b)>=x%;

注：PrbABS6即主邻小区RSRP差值的绝对值小于6的概率。

在根据上述方法计算出的三元组<a, b, c>中，小区a、b、c相互间信号差别不大，没有主导小区，出现严重重叠覆盖，会严重影响通话质量、下载速率等KPI性能指标。

注意：

1. 要求在设计系统时，x参数可配，在界面上输入不同的值显示不同的结果，例如x=70;
2. 求三元组时要求去重。

## 2．系统性能需求分析

为了保证系统能够长期、安全、稳定、可靠的运行，数据库分析管理系统应该满足以下性能需求：

（1）系统的安全性：

① 系统应设置访问用户的标识以鉴别是否是合法的用户，并要求合法用户设置其密码，保证用户身份不被盗用；

② 系统应对不同的数据设置不同的访问级别，限制访问用户可以查询和处理数据；

（2）系统的完整性：

① 各种信息记录的完整性，信息记录不能为空；

② 各种数据间相互作用的联系的正确性；

③ 相同的数据在不同记录中的一致性；

（3）系统的可操作性：本系统采用C/S模式设计，客户端界面简洁明了，易于操作。

# 三、设计

## 1. 层次结构



图1 TD-LTE干扰分析系统层次结构图

人机交互层：人与计算机之间的界面，用户的命令由交互层接受，通过模块间的接口传到逻辑层进行处理，并且在交互层观察执行结果。

业务逻辑层：收到界面层传来的指令后作出相应的逻辑判断，调用对应的模块来处理请求。

数据访问层：用来存储，检索，更新和删除对象，使底层的存储技术不暴露出来。数据层的封装，使得无论存储策略如何变化，业务逻辑层对象都不受影响。

## 2. 业务层软件模块设计（写出每个模块的功能）

### 2.1 系统程序流程图



### 用户管理模块

功能：程序的入口，只有已经注册过的用户才有权限使用该数据库管理程序。

用户注册：



|  |  |  |
| --- | --- | --- |
| 输入 | 处理 | 输出 |
| 1. 申请注册的用户在ui的文本框中输入用户名，密码 2. 读取已注册用户信息记录文件 | 1. 检查新用户的用户名是否存在 2. 若不存在，将密码加密 3. 若已存在，不做处理 | 1. 对于不存在的用户，将用户名和密码写入文件中 2. 对不符合规范的注册行为做出提醒 |

用户登录：



|  |  |  |
| --- | --- | --- |
| 输入 | 处理 | 输出 |
| 1. 读取文本框中的内容 2. 读取已注册用户信息 | 1. 检查是否有输入 2. 若有输入，和注册用户信息比较 | 1. 输入为空，提示请输入信息 2. 密码错误，提示请注册 3. 密码正确，连接数据库，进入系统 |

### 2.3 数据库访问操作

#### 2.3.1导入/导出数据模块

功能：将用户指定的文件导入相对应的数据库表中，也需要把允许导出的数据库表导入指定的excel文件里。

导入数据：



|  |  |  |
| --- | --- | --- |
| 输入 | 处理 | 输出 |
| 1. 每批次导入数据的行数（默认值为50行） 2. 指定被导入数据的路径 | 1. 清洗，把数据类型不相符的内容去除 2. 任务提交给数据库服务器 3. 更新进度条 | 导入进度，插入是否成功都能从进度条中看出来 |

导出数据：



|  |  |  |
| --- | --- | --- |
| 输入 | 过程 | 输出 |
| 待写入文件的路径名 | 1. 读数据库表 2. 创建excel表 3. 将数据写入excel表中 | 通过进度条显示导出进度 |

#### 信息查询模块

功能：允许用户对tbCell，tbPRBnew，tbKPI等数据库表执行查询操作，并且可以根据查到的信息生成直观的图表。

查询KPI表：

1.



|  |  |  |
| --- | --- | --- |
| 输入 | 处理 | 输出 |
| 从输入框中读取网元名称 | 执行sql语句，查出KPI表中的所有属性项  筛选符合要求的属性 | 在UI上显示选出的属性 |

2.



|  |  |  |
| --- | --- | --- |
| 输入 | 处理 | 输出 |
| 1. 通过日历选择时间段 2. 输入或者通过下拉列表选择要查的属性 | 1. 从数据库读取对应信息 2. 将信息转化成图像 | 折线图和各点对应的横纵坐标 |

PRB小时级汇聚



|  |  |  |
| --- | --- | --- |
| 输入 | 处理 | 输出 |
| 用户点击按钮 | 调用数据库的存储过程 | 表的内容更新（用户不可见） |

PRB查询

1.



|  |  |  |
| --- | --- | --- |
| 输入 | 处理 | 输出 |
| 从输入框中读取网元名称 | 执行sql语句，查出PRB表中的所有属性项  筛选符合要求的属性 | 在UI上显示选出的属性 |

2.



|  |  |  |
| --- | --- | --- |
| 输入 | 处理 | 输出 |
| 1. 通过日历选择时间段 2. 输入或者通过下拉列表选择要查的属性 | 1. 从数据库读取对应信息 2. 将信息转化成图像 | 折线图和各点对应的横纵坐标 |

#### 干扰分析模块

功能：对tbMROData表进行统计，生成两张新的数据库表供用户查看。

C2I分析：



|  |  |  |
| --- | --- | --- |
| 输入 | 处理 | 输出 |
| 从对话框中获取用户指定的主小区和干扰小区最小对数 | 调用数据库中写好的存储过程 | 生成新表tbC2Inew投到ui中 |

分析三元组：



|  |  |  |
| --- | --- | --- |
| 输入 | 处理 | 输出 |
| 由用户输入主邻小区对的阈值参数 | 调用数据库存储过程 | 生成新表tbC2Inew投到UI中 |

导出图表：



|  |  |  |
| --- | --- | --- |
| 输入 | 过程 | 输出 |
| 由用户自选保存路径 | 在指定位置创建文件并保存 | 提示用户保存成功 |

## 3. 数据库层设计

### 3.1 E-R图



### 3.2 数据库关系表设计

一个实体型转换为一个关系模式，实体的属性就是关系的属性，实体的关键字就是关系的关键字。一个1:1的联系可以转换为一个独立的关系模式，也可以与任意一端对应的关系模式合并，这次的设计中没有1:1的联系，全部都是多对多的联系。而一个m:n的联系必须转换成一个独立的关系模式，与该联系相连的各实体的键以及联系本身的属性均转换为关系模式的属性，而关系模式的键为实体键的组合。

关系表定义如下：

tbCell([CITY],**[SECTOR\_ID],**[SECTOR\_NAME],[ENODEBID],[ENODEB\_NAME],[EARFCN],[PCI],[PSS],[SSS],[TAC],[VENDOR],[LONGITUDE], [LATITUDE], [STYLE],[AZIMUTH], [HEIGHT],[ELECTTILT],[MECHTILT],[TOTLETILT])

tbAdjCell(**[S\_SECTOR\_ID]**,**[N\_SECTOR\_ID]**,[S\_EARFCN],[N\_EARFCN])

tbSecAdjCell(**S\_SECTOR\_ID,N\_SECTOR\_ID**)

tbATUC2I(**SECTOR\_ID,NCELL**\_ID,RATIO\_ALL,RANK,COSITE)

tbOptCell(**SECTOR\_ID**，EARFCN，CELL\_TYPE)

tbPCIAssigment(**ASSIGN\_ID**，EARFCN，SECTOR\_ID，SECTOR\_NAME，ENODEB\_ID，PCI，PSS，SSS，LONGITUDE，LATITUDE，STYLE，OPT\_DATETIME)

tbATUData(**seq**，FileName Time，Longitude，Latitude，CellID，TAC，EARFCN，PCI，RSRP，RS，SINR，NCell\_ID\_1，NCell\_EARFCN\_1，NCell\_PCI\_1，NCell\_RSRP\_1，NCell\_ID\_2，NCell\_EARFCN\_2，NCell\_PCI\_2，NCell\_RSRP\_2，NCell\_ID\_3，NCell\_EARFCN\_3，NCell\_PCI\_3，NCell\_RSRP\_3，NCell\_ID\_4，NCell\_EARFCN\_4，NCell\_PCI\_4，NCell\_RSRP\_4，NCell\_ID\_5，NCell\_EARFCN\_5，NCell\_PCI\_5，NCell\_RSRP\_5，NCell\_ID\_6，NCell\_EARFCN\_6，NCell\_PCI\_6，NCell\_RSRP\_6)

tbMROData(**TimeStamp，ServingSector，InterferingSector**，LteScRSRP，LteNcRSRP，LteNcEarfcn，LteNcPci)

tbATUHandOver(**SSECTOR\_ID,NSECTOR\_ID**,HOATT)

tbC2I(CITY,**SCELL,NCELL**,PrC2I9,C2I\_Mean,std,SampleCount,WeightedC2I)

tbHandOver(CITY,**SCELL,NCELL**,HOATT,HOSUCC,HOSUCCRATE)

tbKPI(**[起始时间]**,[周期],[网元名称],[基站],**[小区]**,[RRC连接建立完成次数 (无)],[RRC连接请求次数（包括重发） (无)],[RRC建立成功率qf (%)],[E-RAB建立成功总次数 (无)],[E-RAB建立尝试总次数 (无)],[E-RAB建立成功率2 (%)],[eNodeB触发的E-RAB异常释放总次数 (无)],[小区切换出E-RAB异常释放总次数 (无)],[E-RAB掉线率(新) (%)],[无线接通率ay (%)],[eNodeB发起的S1 RESET导致的UE Context释放次数 (无)],[UE Context异常释放次数 (无)],[UE Context建立成功总次数 (无)] ,[无线掉线率 (%)],[eNodeB内异频切换出成功次数 (无)],[eNodeB内异频切换出尝试次数 (无)],[eNodeB内同频切换出成功次数 (无)] ,[eNodeB内同频切换出尝试次数 (无)],[eNodeB间异频切换出成功次数 (无)],[eNodeB间异频切换出尝试次数 (无)],[eNodeB间同频切换出成功次数 (无)],[eNodeB间同频切换出尝试次数 (无)],[eNB内切换成功率 (%)],[eNB间切换成功率 (%)],[同频切换成功率zsp (%)],[异频切换成功率zsp (%)],[切换成功率 (%)],[小区PDCP层所接收到的上行数据的总吞吐量 (比特),[小区PDCP层所发送的下行数据的总吞吐量 (比特)],[RRC重建请求次数 (无)],[RRC连接重建比率 (%)],[通过重建回源小区的eNodeB间同频切换出执行成功次数 (无)] [通过重建回源小区的eNodeB间异频切换出执行成功次数 (无)],[通过重建回源小区的eNodeB内同频切换出执行成功次数 (无)],[通过重建回源小区的eNodeB内异频切换出执行成功次数 (无)],[eNB内切换出成功次数 (次),[eNB内切换出请求次数 (次)])

tbPRB(**[起始时间]**,[周期],[网元名称],[小区],**[小区名]**,[第0个PRB上检测到的干扰噪声的平均值 (毫瓦分贝)],[第1个PRB上检测到的干扰噪声的平均值 (毫瓦分贝)],[第2个PRB上检测到的干扰噪声的平均值 (毫瓦分贝)],[第3个PRB上检测到的干扰噪声的平均值 (毫瓦分贝)],[第4个PRB上检测到的干扰噪声的平均值 (毫瓦分贝)],[第5个PRB上检测到的干扰噪声的平均值 (毫瓦分贝)],[第6个PRB上检测到的干扰噪声的平均值 (毫瓦分贝)],[第7个PRB上检测到的干扰噪声的平均值 (毫瓦分贝)],[第8个PRB上检测到的干扰噪声的平均值 (毫瓦分贝)],[第9个PRB上检测到的干扰噪声的平均值 (毫瓦分贝)],[第10个PRB上检测到的干扰噪声的平均值 (毫瓦分贝)],……,[第90个PRB上检测到的干扰噪声的平均值 (毫瓦分贝)],[第91个PRB上检测到的干扰噪声的平均值 (毫瓦分贝)],[第92个PRB上检测到的干扰噪声的平均值 (毫瓦分贝)],[第93个PRB上检测到的干扰噪声的平均值 (毫瓦分贝)],[第94个PRB上检测到的干扰噪声的平均值 (毫瓦分贝)],[第95个PRB上检测到的干扰噪声的平均值 (毫瓦分贝)],[第96个PRB上检测到的干扰噪声的平均值 (毫瓦分贝)],[第97个PRB上检测到的干扰噪声的平均值 (毫瓦分贝)],[第98个PRB上检测到的干扰噪声的平均值 (毫瓦分贝)],[第99个PRB上检测到的干扰噪声的平均值 (毫瓦分贝)])

由于关系模式都是不可再分的数据项，很显然为第一范式，但tbPRB中网元名部分依赖于主键<起始时间，小区名>所以低于第二范式。

## 4. 数据库文件/文件组设计

在Microsoft sql server 2014中建立TD\_LTE数据库，相关设计如下：

数据库主文件：TD\_LTE\_m.mdf

数据库辅文件：TD\_LTE\_n1.ndf、TD\_LTE\_n2.ndf和TD\_LTE\_n3.ndf

日志文件：TD\_LTE\_log.ldf

文件组：TD\_LTE\_G1

将文件组TD\_LTE\_G1安排在不同磁盘分区：

在D、E、F三个磁盘驱动器上创建三个文件TD\_LTE\_n1.ndf、TD\_LTE\_n2.ndf和TD\_LTE\_n3.ndf，然后将它们分配给文件组TD\_LTE\_G1。脚本如下：

CREATE DATABASE TD\_LTE

ON PRIMARY

(

NAME = TD\_LTE\_m,--主文件

FILENAME = 'E:\sql server management studio\TD\_LTE\_m.mdf',

SIZE = 800,

MAXSIZE = 1000,

FILEGROWTH = 40

),

filegroup TD\_LTE\_G1

(

NAME = TD\_LTE\_n1,--辅文件

FILENAME = 'D:\sql server management studio\TD\_LTE\_n1.ndf',

SIZE = 800,

MAXSIZE = 1000,

FILEGROWTH = 40

),

(

name = TD\_LTE\_n2, --辅文件2

filename = 'E:\sql server management studio\TD\_LTE\_n2.ndf',

size=1MB,

maxsize=40MB,

filegrowth=2MB

),

(

name = TD\_LTE\_n3, --辅文件3

filename = 'F:\sql server management studio\TD\_LTE\_n3.ndf',

size=1MB,

maxsize=40MB,

filegrowth=2MB

)

LOG ON ---日志文件

(

name = 'TD\_LTE\_log',

filename = 'D:\sql server management studio\TD\_LTE\_log.ldf',

size=1MB,

maxsize=10MB,

filegrowth=1MB

);

## 5. 索引设计

### 5.1 聚集索引设计

在表tbCell的SECTOR\_ID属性上建立索引sectorIndex：

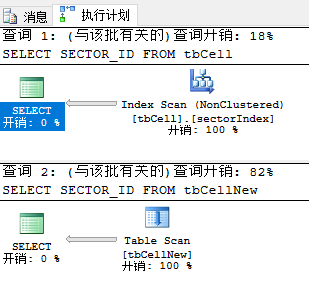
create index sectorIndex on tbCell(SECTOR\_ID)

分别在表tbCell及其备份表tbCellNew上执行同样的查询，观察执行计划：

SELECT SECTOR\_ID FROM tbCell

SELECT SECTOR\_ID FROM tbCellNew

执行计划对比截图：



说明SECTOR\_ID属性上聚集索引的存在提高了查询效率。

### 5.2 非聚集索引的创建

(1). 在表tbKPI的属性‘网元名称’上建立非聚集索引inx\_wangyuan

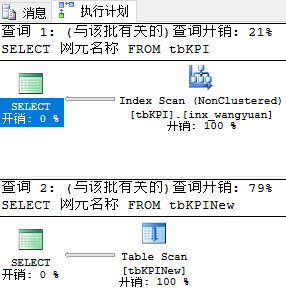
create nonclustered index inx\_wangyuan on tbKPI(网元名称)

分别在表tbKPI及其备份表tbKPINew上执行查询需求完全一样两条select语句：

SELECT 网元名称 FROM tbKPI

SELECT 网元名称 FROM tbKPINew

查看执行计划：



可知使用非聚集索引的查询提高了查询效率，该非聚集索引是有效的。

1. 同理，由（1）可知，在表tbPRB的属性‘网元名称’上建立非聚集索引inx\_wangyuan

create nonclustered index inx\_wangyuan on tbprb(网元名称)

1. 在表tbCell的属性‘ENODEBID’上建立非聚集索引sectorENODEBID

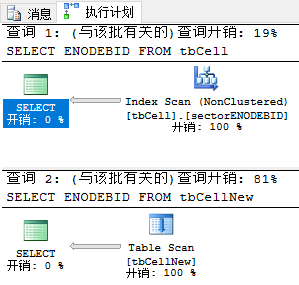
create nonclustered index sectorENODEBID on tbCell(ENODEBID)

分别在表tbCell及其备份表tbCellNew上执行同样的查询，观察执行计划：

SELECT ENODEBID FROM tbCell

SELECT ENODEBID FROM tbCellNew

执行计划对比截图：



说明该非聚集索引提高了查询效率。

1. 在表tbMROdata的属性‘TIMESTAMP’上建立非聚集索引inx\_time

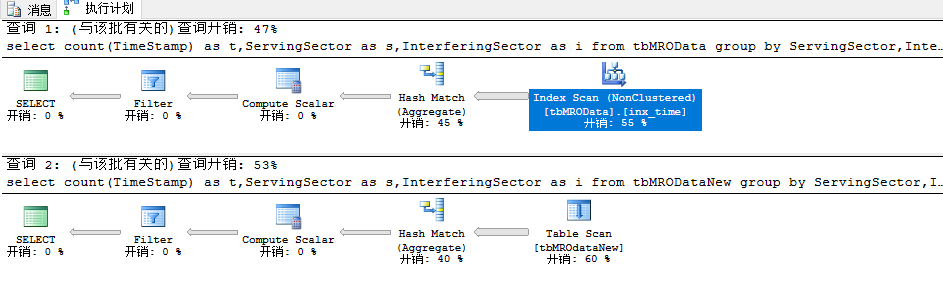
create nonclustered index inx\_time on tbMroDATA (TIMESTAMP)

分别在表tbMROData及其备份表tbMRODataNew上执行两条查询需求完全一样的2条select语句，观察执行计划：

select count(TimeStamp) as t,ServingSector as s,InterferingSector as i from tbMROData group by ServingSector,InterferingSector having count(TimeStamp)>10

select count(TimeStamp) as t,ServingSector as s,InterferingSector as i from tbMRODataNew group by ServingSector,InterferingSector having count(TimeStamp)>10

执行计划对比截图：



可知使用非聚集索引的查询提高了查询效率，该非聚集索引是有效的。

## 6. 存储过程和触发器

### 6.1数据管理

#### 6.1.1 数据插入

向数据库中的1个关系表中导入一条主键值不为空的数据（对应于一个元组）时，根据该条数据中的主属性/主键，判断本条数据是否已在数据库表中；如果不在，则将该条数据作为新数据插入数据库。如果数据已经存在，则删除数据库中已有数据，将该条数据插入数据库，即用新数据覆盖数据库中的已有旧数据。

以tbCell表为例，当有逐渐重复的数据时其更新的方法为删除原有内容，并插入一条新的，其触发器的内容为：

ALTER TRIGGER [dbo].[trigger1]

ON [dbo].[tbCell]

INSTEAD OF insert

AS

declare @C nvarchar(255)declare @ID nvarchar(50)

declare @SN nvarchar(255)declare @EID INT

declare @EN NVARCHAR(255)declare @EA INT

declare @PCI INTdeclare @PSS INT

declare @SSS INTdeclare @TAC INT

declare @VD NVARCHAR(255)declare @LO REAL

declare @LA REALdeclare @SY NVARCHAR(255)

declare @AZ FLOATdeclare @HE FLOAT

declare @EL FLOATdeclare @ME FLOATdeclare @TO FLOAT

select @C=CITY, @ID=SECTOR\_ID, @SN=SECTOR\_NAME,@EID =ENODEBID, @EN =ENODEB\_NAME,@EA =EARFCN, @PCI =PCI,@PSS =PSS,@SSS=SSS, @TAC=TAC,@VD =VENDOR,@LO=LONGITUDE,@LA =LATITUDE,@SY =STYLE,@AZ =AZIMUTH, @HE =HEIGHT, @EL =ELECTTILT,@ME =MECHTILT,@TO=TOTLETILT from inserted

if((select count(\*) from tbCell where SECTOR\_ID=@ID )>0)

begin

delete from tbCell where (SECTOR\_ID=@ID

INSERT INTO tbCell VALUES (@C ,@ID ,@SN ,@EID , @EN ,@EA ,@PCI ,@PSS ,@SSS ,@TAC ,@VD ,@LO , @LA ,@SY , @AZ , @HE ,@EL ,@ME ,@TO)

end

ELSE

INSERT INTO tbCell VALUES (@C ,@ID ,@SN ,@EID , @EN ,@EA ,@PCI ,@PSS ,@SSS ,@TAC ,@VD ,@LO , @LA ,@SY , @AZ , @HE ,@EL ,@ME ,@TO)

其中INSTEAD OF insert是将插入操作替换为触发器定义的行为，不会出现主键重复的报错，而用for insert时会先执行insert然后再执行触发器，这时已经发生了主键冲突的错误，触发器也就失去了它的作用。

#### 6.1.2 数据更新

用更新的方法来插入新数据的触发器为：

ALTER TRIGGER [dbo].[trigger2]

ON [dbo].[tbCell]

INSTEAD OF insert

AS

declare @C nvarchar(255)declare @ID nvarchar(50)

declare @SN nvarchar(255)declare @EID INT

declare @EN NVARCHAR(255)declare @EA INT

declare @PCI INTdeclare @PSS INT

declare @SSS INTdeclare @TAC INT

declare @VD NVARCHAR(255)declare @LO REAL

declare @LA REALdeclare @SY NVARCHAR(255)

declare @AZ FLOATdeclare @HE FLOAT

declare @EL FLOATdeclare @ME FLOATdeclare @TO FLOAT

select @C=CITY, @ID=SECTOR\_ID, @SN=SECTOR\_NAME,@EID =ENODEBID, @EN =ENODEB\_NAME,@EA =EARFCN, @PCI =PCI,@PSS =PSS,@SSS=SSS, @TAC=TAC,@VD =VENDOR,@LO=LONGITUDE,@LA =LATITUDE,@SY =STYLE,@AZ =AZIMUTH, @HE =HEIGHT, @EL =ELECTTILT,@ME =MECHTILT,@TO=TOTLETILT from inserted

if((select count(\*) from tbCell where SECTOR\_ID=@ID )>0)

begin

UPDATE tbCell SET CITY=@C, SECTOR\_ID=@ID, SECTOR\_NAME=@SN,ENODEBID=@EID , ENODEB\_NAME=@EN ,EARFCN=@EA, PCI =@PCI,PSS =@PSS,SSS=@SSS, TAC=@TAC,VENDOR=@VD ,LONGITUDE=@LO,LATITUDE=@LA ,STYLE=@SY ,AZIMUTH=@AZ ,HEIGHT= @HE , ELECTTILT=@EL,MECHTILT=@ME,TOTLETILT=@TO WHERE SECTOR\_ID=@ID

end

ELSE

INSERT INTO tbCell VALUES (@C ,@ID ,@SN ,@EID , @EN ,@EA ,@PCI ,@PSS ,@SSS ,@TAC ,@VD ,@LO , @LA ,@SY , @AZ , @HE ,@EL ,@ME ,@TO)

大部分内容并没有变，只是将删除和插入的两条语句换成了update。效率更高了。

### 6.2 数据查询

#### 6.2.1 小区配置信息查询

从tbCell表中查询小区信息。

用存储过程实现

ALTER proc [dbo].[cellProc]

@cname nvarchar(100)=NULL,

@cid nvarchar(100)=NULL

as

begin

select \* from tbcell where SECTOR\_NAME=@cname or SECTOR\_ID=@cid

end

需要的参数为小区名SECTOR\_NAME或SECTOR\_ID二者之一就可以。



#### 6.2.2 基站eNodeB信息查询

和上面获取小区配置信息相似

ALTER proc [dbo].[enodeProc]

@ename nvarchar(100),

@eid int

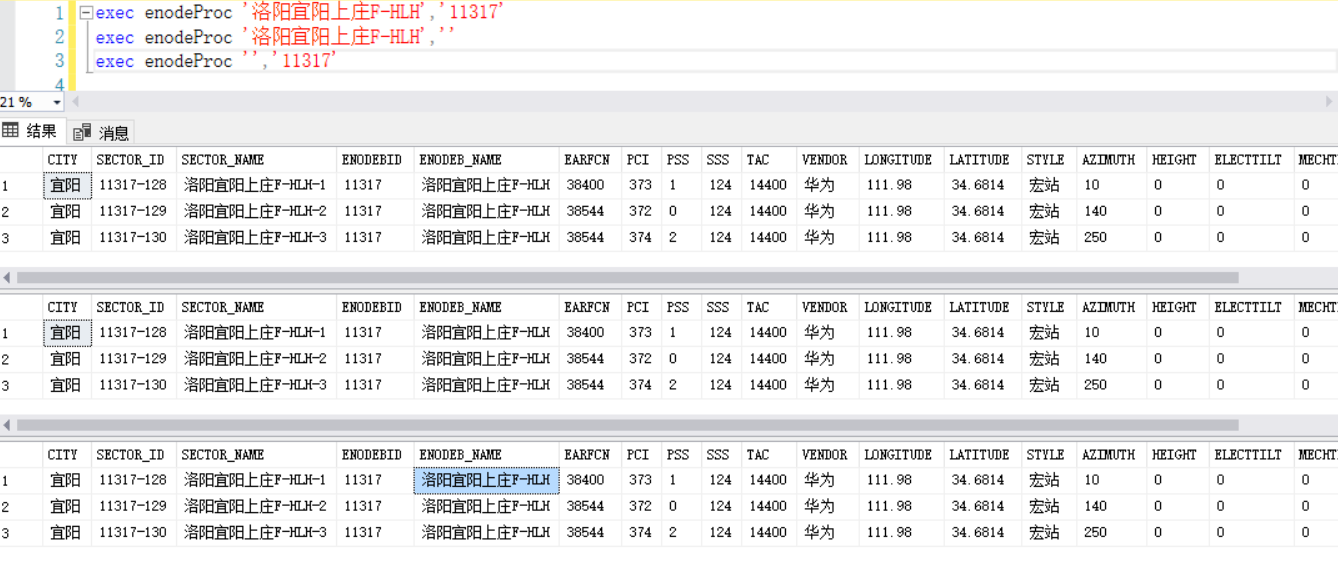
as

begin

select SECTOR\_NAME, SECTOR\_ID from tbcell where ENODEB\_NAME=@ename or ENODEBID=@eid

end

这个查询只要求显示属于基站的小区，所以不用把相关小区的所有属性全都搜出来。



#### 6.2.3 KPI指标信息查询

这一部分需要指定好要查询的网元，因为网元不是主键，所以设计的存储过程是把数据按网元分组，再计算同一网元的各个小区的平均数，查询的时间段和属性使由用户指定的：

ALTER PROCEDURE [dbo].[KPI2]

@NAME NVARCHAR(100),

@sDATE NVARCHAR(100),

@eDATE NVARCHAR(100),

@ATTR NVARCHAR(255)

AS

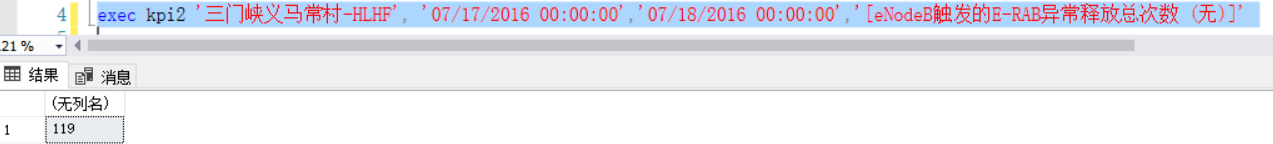
BEGIN

declare @sql nvarchar(300)

set @sql='SELECT avg( '+@ATTR+') FROM tbKPI group by 网元名称,起始时间 having 网元名称='''+@NAME+'''and(substring(起始时间,1,10) BETWEEN '''+@sDATE+'''AND'''+@eDATE+''')'

exec(@sql)

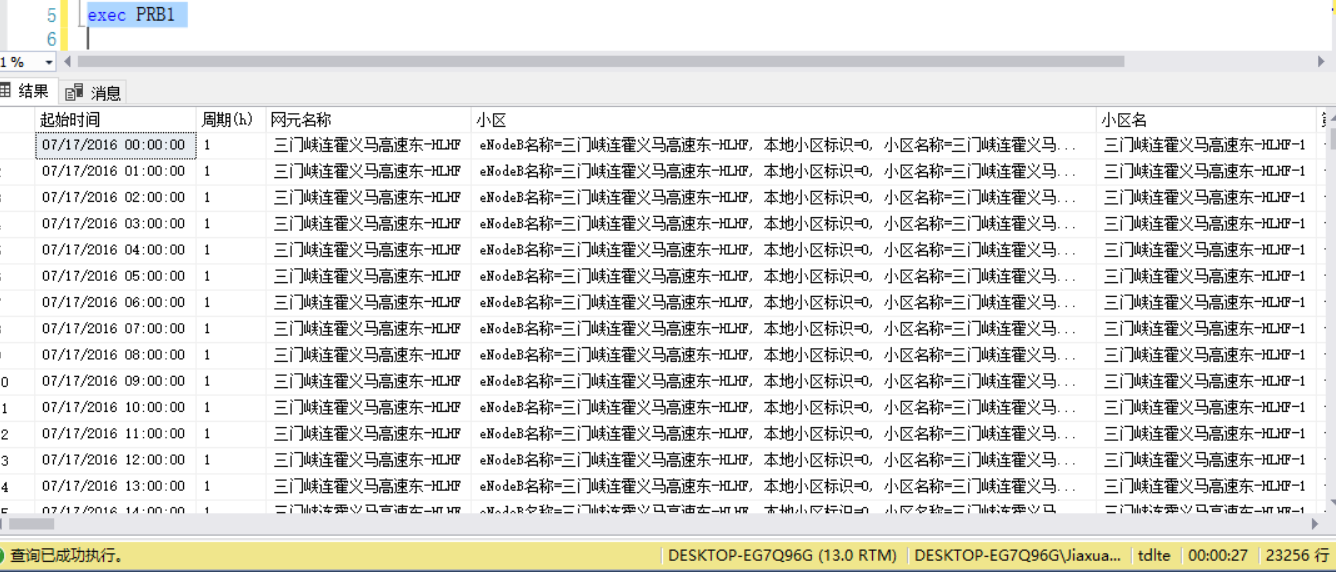
end



#### 6.2.4 PRB信息统计与查询

根据表“优化区17日-19日每PRB干扰 查询-15分钟”，统计小时级的PRB干扰数据，生成一张新表tbPRBnew，并导出到外部的excel表中。某一网元某一小时内的PRB干扰=该小时内4个15分钟平均值的和/4，要求计算所有的网元从2016年7月17日到7月19日的每个小时的100个PRB上检测到的干扰噪声的平均值。

因为要将9万多行的数据进行统计并导入新表，所以耗费的时间比较长。一般情况下大约需要40s。



这一部分的代码挺长的但是过程并不复杂，先声明一个表变量用来存储原来的tbPRB，然后把这个表变量按起始时间和小区名排序，每次取第一行给变量赋值，之后删掉这一行，当行数到4时，将取到的数据整理后插入tbPRBnew中，再将变量复原准备接受下面4行的数据。

之后在对tbPRBnew分析，根据（1）中得到的小时级的PRB干扰数据表，在界面上根据网元名称以输入框或者下拉列表的方式选定某个特定网元，在右侧显示其属性列表，添加时间控件，查询网元某个时间段（小时级）某个属性值的变化情况

ALTER PROCEDURE [dbo].[PRB3]

@NAME NVARCHAR(100),

@sDATE NVARCHAR(100),

@eDATE NVARCHAR(100),

@ATTR NVARCHAR(255)

AS

BEGIN

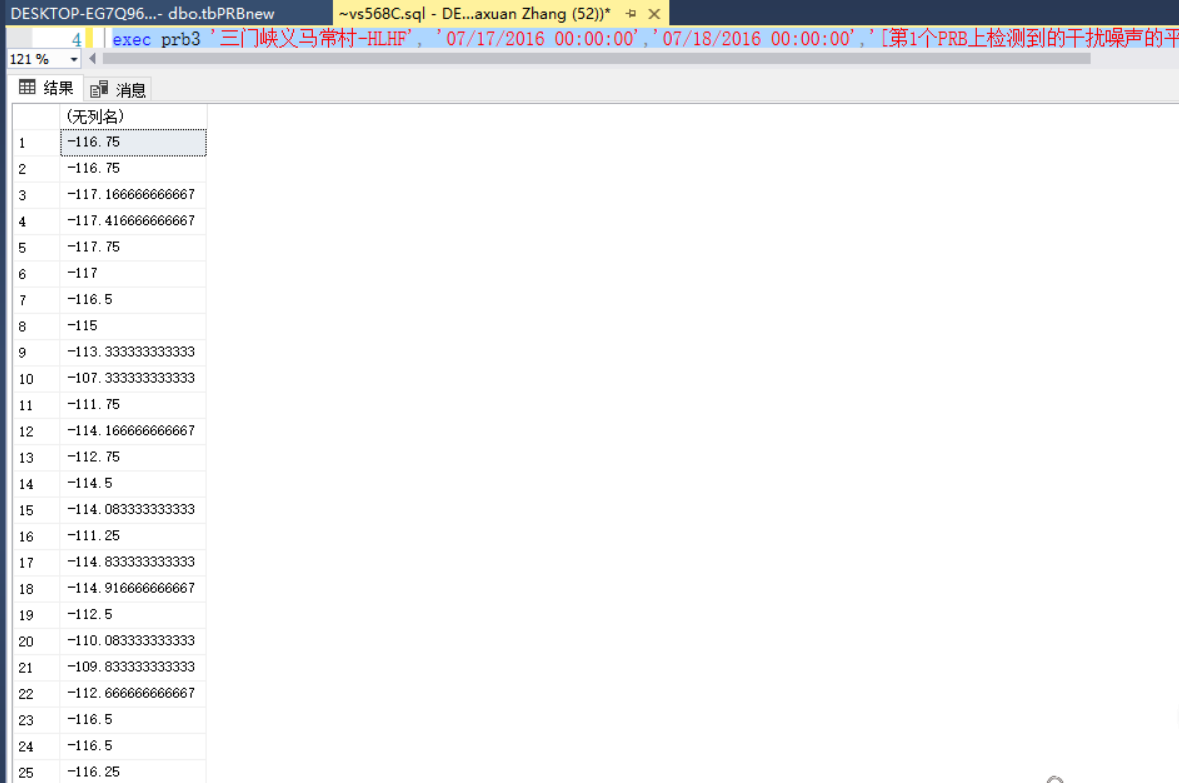
declare @sql nvarchar(300)

set @sql='SELECT avg('+@ATTR+') FROM tbPRBnew group by 网元名称,起始时间 having 网元名称='''+@NAME+'''and(起始时间 BETWEEN '''+@sDATE+'''AND'''+@eDATE+''')'

print @sql

exec(@sql)

end



方法和查询KPI时的大同小异。

### 6.3 数据分析

#### 6.3.1 主邻小区C2I干扰分析

具体要求和实现步骤如下：

1．主邻小区RSRP差值（即C2I）的均值C2I\_mean、标准差std计算

根据导入的MRO测量报告数据表tbMROData表中主小区和邻小区的参考信号接收功率RSRP，计算主邻小区RSRP差值，该联系是多对多联系。

计算每对主邻小区<SCell, NCell>的RSRP差值C2I，并计算出C2I的均值和标准差。

2．主邻小区RSRP差值小于9的概率PrbC2I9计算

根据每对主邻小区RSRP差值的均值和标准差，可以得到正态分布，根据正态分布求出主邻小区RSRP差值小于9的概率Prb9=Pr{C2I<9db}。

3．主邻小区RSRP差值绝对值小于6的概率PrbABS6的计算

根据每对主邻小区RSRP差值的均值和标准差，可以得到正态分布，根据正态分布求出主邻小区RSRP差值绝对值小于6的概率（PrbABS6）。

4．生成新表tbC2Inew

新表包括六个属性：主小区ID、邻小区ID、主邻小区RSRP差值的均值、主邻小区RSRP差值的标准差、主邻小区RSRP差值小于9的概率、主邻小区RSRP差值绝对值小于6的概率。

ALTER PROCEDURE [dbo].[c2i]

@n int

AS

BEGIN

IF EXISTS(SELECT \* FROM sys.Tables WHERE name='tbC2Inew')

DROP TABLE tbC2Inew;

with temp as(

select count(TimeStamp) as t,ServingSector as s,InterferingSector as i from tbMROData group by ServingSector,InterferingSector having count(TimeStamp)>@n),

TEMP1 AS(select TOP(9000) ServingSector,InterferingSector, avg(LteScRSRP-LteNcRSRP) AS MEAN,STDEV(LteScRSRP-LteNcRSRP)AS STD from tbMROData GROUP BY ServingSector,InterferingSector having count(TimeStamp)in(select t from temp) order by ServingSector,InterferingSector)

SELECT ServingSector,InterferingSector,MEAN,STD,dbo.dist((9-MEAN)/STD)as PrC2I9,dbo.dist((6-MEAN)/STD)-dbo.dist((-6-MEAN)/STD)as PrbABS6 into tbC2Inew FROM TEMP1

select \* from tbc2inew

end

两个临时表的作用是筛选出< ServingSector,InterferingSector>大于规定下限的主邻小区，第二个临时表能计算出RSRP的差值的均值和标准差，但是要计算概率分布时发现sql没有提供内聚函数，所以需要自己编写，我是参照C#代码改写成sql函数的。估算公式为

ALTER function [dbo].[dist]( @x float)

returns float

as

begin

declare @t table(id float(8),rownum int)

insert into @t values(0.196854,1)

insert into @t values(0.115194,2)

insert into @t values(0.000344,3)

insert into @t values(0.019527,4)

declare @m float(8)=0

select @m=id from @t where rownum=3

declare @x0 float=@x

if @x0<0

begin

set @x0=-@x0

end

declare @erf float(8) =0

declare @i int =0

while @i<4

begin

set @i+=1

select @erf += id \* Power(@x0, @i) from @t where rownum=@i

end

set @erf = 1 - Power(1.0 + @erf, -4)

declare @phi float(8)= 0.5 \* (1 + @erf)

if @x<0

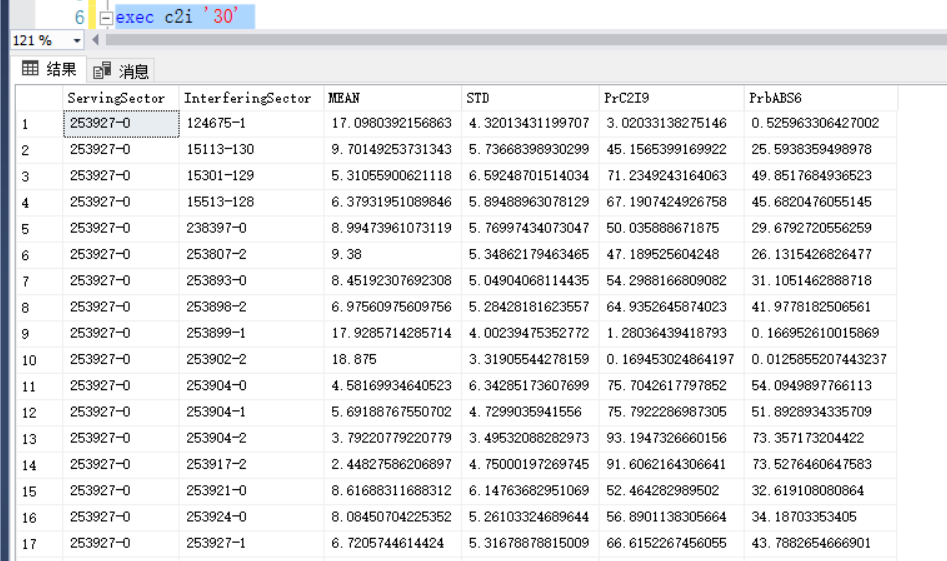
begin

set @phi=0.5 \* (1-@erf)

end

return @phi\*100

end



#### 6.3.2 重叠覆盖干扰三元组

根据表tbC2Inew，找出所有的小区三元组<a,b,c>（其中a,b,c互为邻小区），生成新表tbC2I3，其中有三个属性，分别是三个小区的小区ID。

ALTER PROC [dbo].[C2I2]

@X FLOAT

AS

BEGIN

IF EXISTS(SELECT \* FROM sys.Tables WHERE name='tbC2I3')

DROP TABLE tbC2I3;

create table tbC2I3( CELLa NVARCHAR(100) NOT NULL, CELLb NVARCHAR(100) NOT NULL,CELLc NVARCHAR(100) NOT NULL)

declare @i int=0declare @j int=0;

WITH TEMP AS(SELECT ServingSector, InterferingSector,PrbABS6 FROM tbC2Inew where InterferingSector IN (SELECT ServingSector FROM tbC2Inew)AND PrbABS6>2),

temp1 as(SELECT TEMP.PrbABS6, temp.ServingSector as cell1, temp.InterferingSector as cell2,tbC2Inew.ServingSector as cell4,tbC2Inew.InterferingSector as cell3 FROM TEMP left join tbC2Inew on TEMP.InterferingSector=tbC2Inew.ServingSector AND tbC2Inew.PrbABS6>@X)

select cell1,cell2,cell3,TEMP.ServingSector AS ecell INTO #TEMP2 from temp1 left join TEMP on temp1.cell3=TEMP.ServingSector and temp.InterferingSector=temp1.cell1 and TEMP.PrbABS6>@X

SELECT @i=COUNT(\*)FROM #TEMP2

DECLARE @C1 NVARCHAR(50)=NULLDECLARE @C2 NVARCHAR(50)=NULL

DECLARE @C3 NVARCHAR(50)=NULLDECLARE @C4 NVARCHAR(50)=NULL

while @j<@i

begin

SELECT TOP(1) @C1=cell1,@C2=cell2,@C3=cell3,@C4=ecell FROM #TEMP2

delete top(1) from #TEMP2

if @C4 IS NOT null and not exists (select \* from tbC2I3 where (CELLa=@C1 and CELLb=@C2 and CELLc=@C3)or (CELLa=@C1 and CELLb=@C3 and CELLc=@C2)or (CELLa=@C2 and CELLb=@C1 and CELLc=@C3)or (CELLa=@C2 and CELLb=@C3 and CELLc=@C1)or (CELLa=@C3 and CELLb=@C1 and CELLc=@C2)or (CELLa=@C3 and CELLb=@C2 and CELLc=@C3))

BEGIN

insert into tbC2I3 values(@C1,@C2,@C3)

END

SET @j+=1

end

drop table #TEMP2

select \* from tbC2I3

END

两个临时表用来筛选三元组中的另外两个，应为他们两个要满足他的主小区是另外一对的干扰小区，选完之后两两合并，在去掉重复的元组就形成了满足要求的新表。



## 7. 数据库访问接口

软件模块使用的sql server数据库的API访问接口是ODBC。

# 三、系统实现

## 软硬件平台

Windows 10操作系统下Qt 5.9图形化库

## 开发环境/语言

操作系统：Windows 10

IDE：Qt Creator 4.4.1

语言：C++

## 数据库平台

Microsoft SQL Server 2014

# 四、运行实例

## 4.1 系统登录与用户注册

登陆界面：



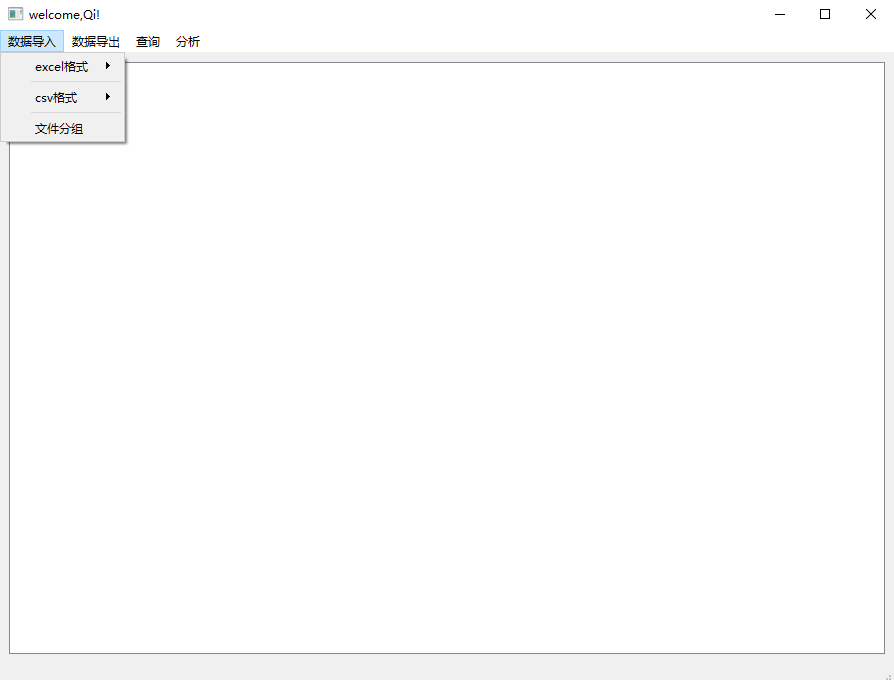
用户注册界面：



## 4.2 数据管理

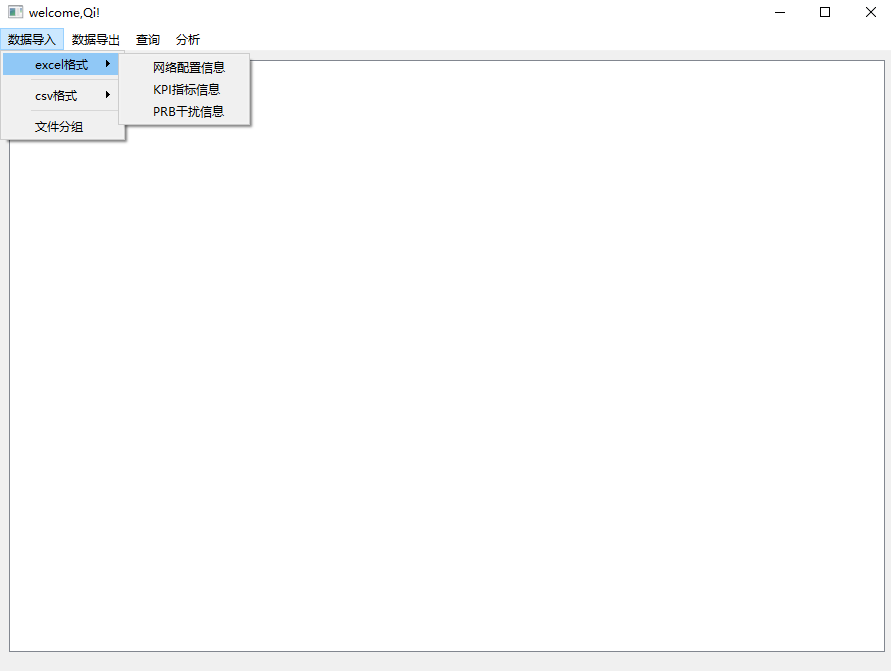
### 4.2.1数据导入

同时支持excel格式和csv格式的文件导入，支持自定义导入文件的分组大小：

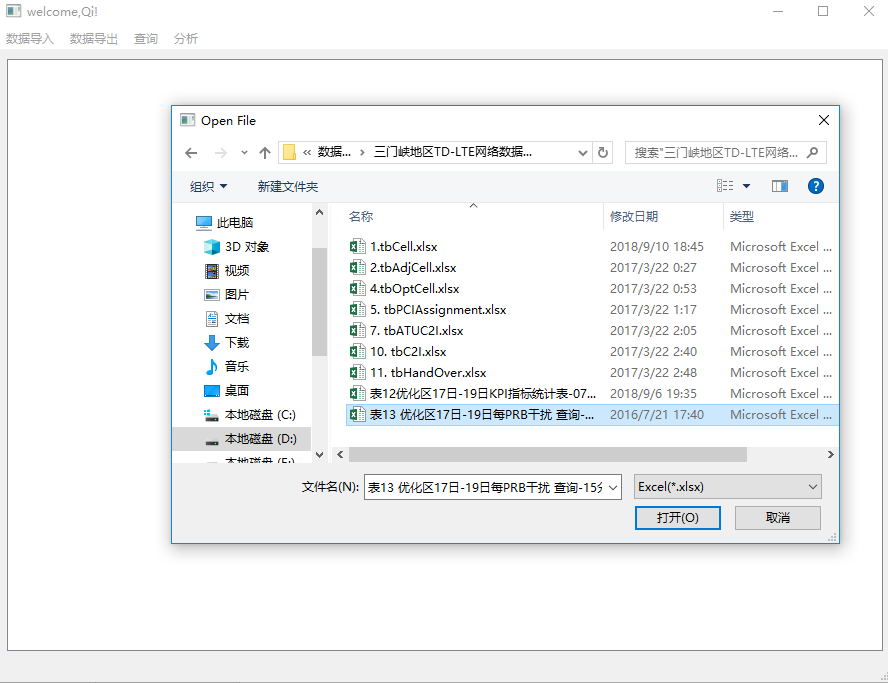


导入excel文件示例：

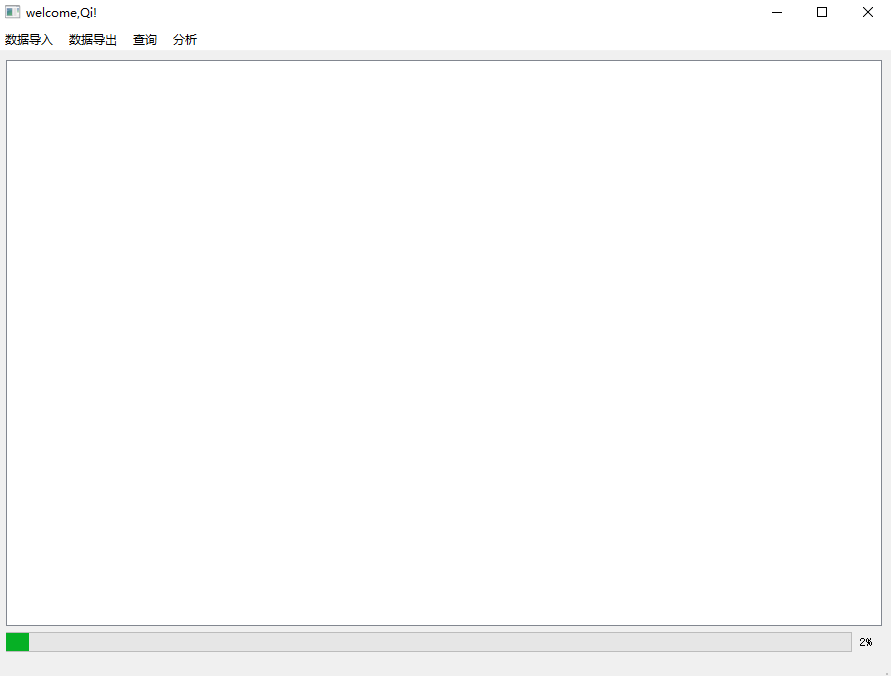
选择要导入的数据库表：



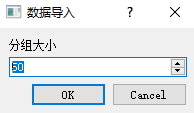
选择要导入的excel格式或csv格式的文件：



导入过程中有进度条提示导入进度：

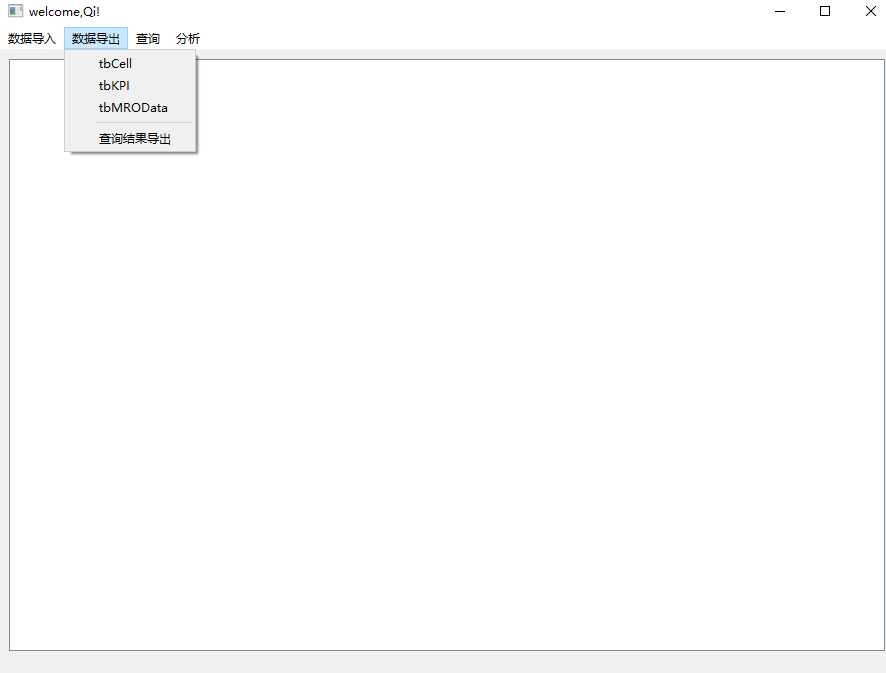


自定义导入时的文件分组大小，默认为50：



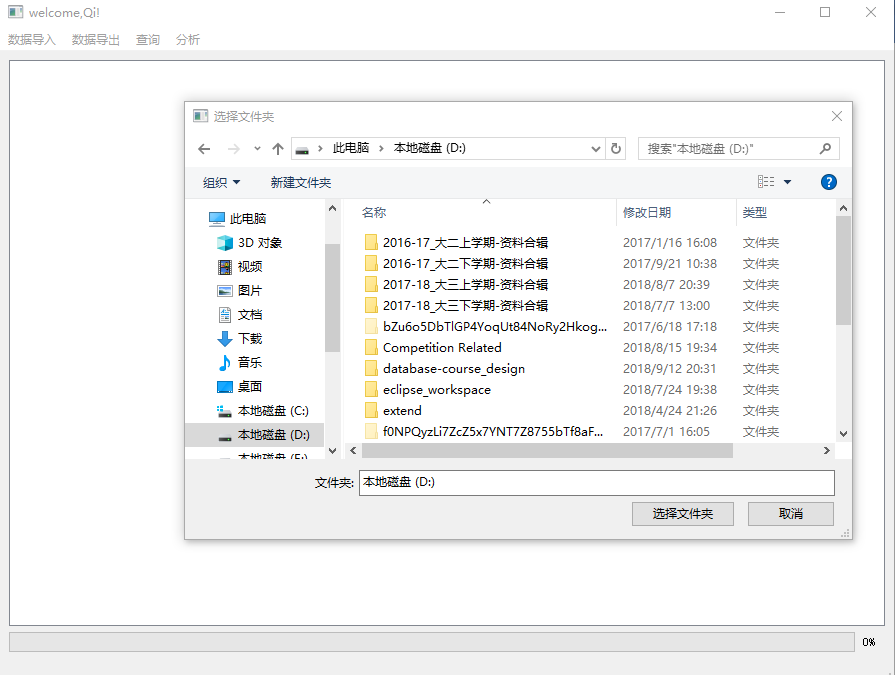
### 4.2.2数据导出

数据导出功能包括数据库表导出和查询结果的导出：

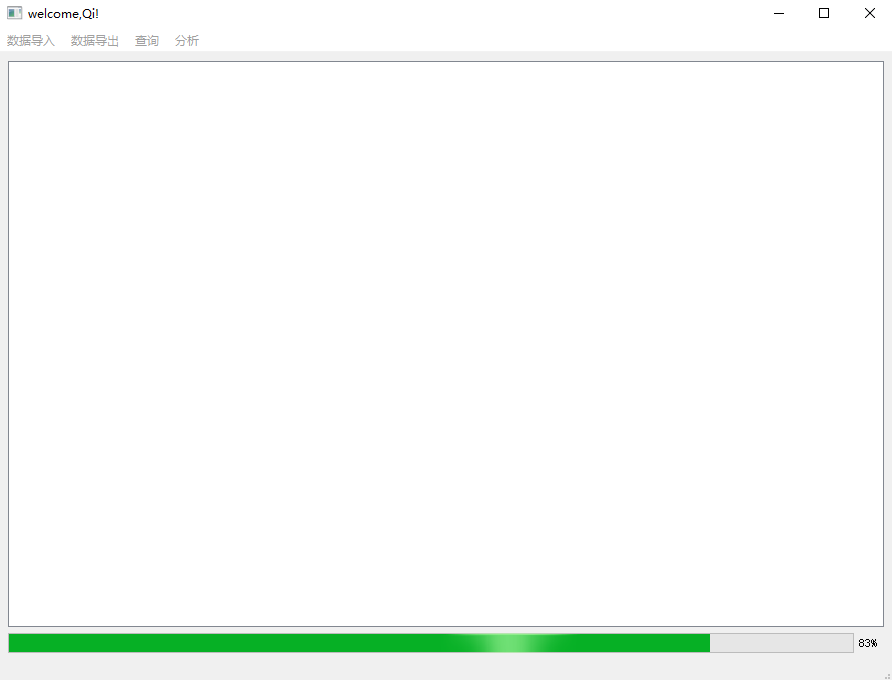


数据库表导出示例：

可以自定义导出路径：

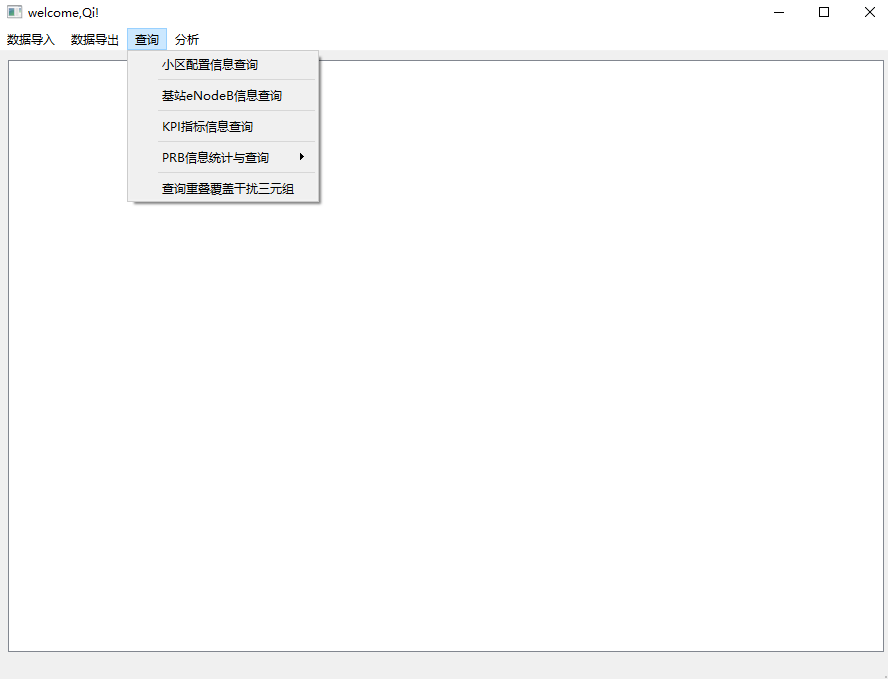


进度条可动态显示导出的进度：



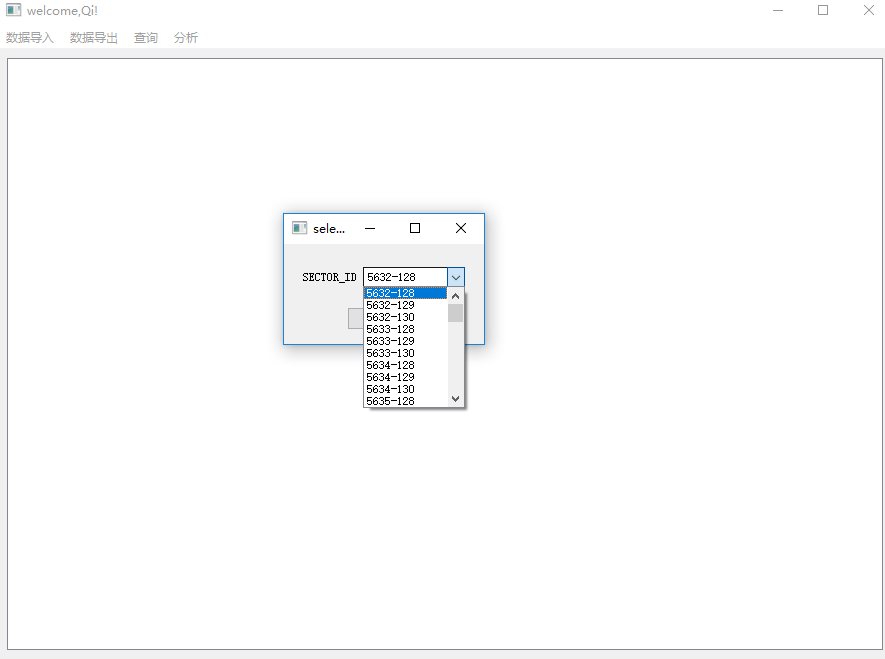
## 信息查询

可查询的信息有小区配置信息查询、基站eNodeB信息查询、KPI指标信息查询、PRB信息统计与查询、重叠覆盖干扰三元组查询。如下图所示：

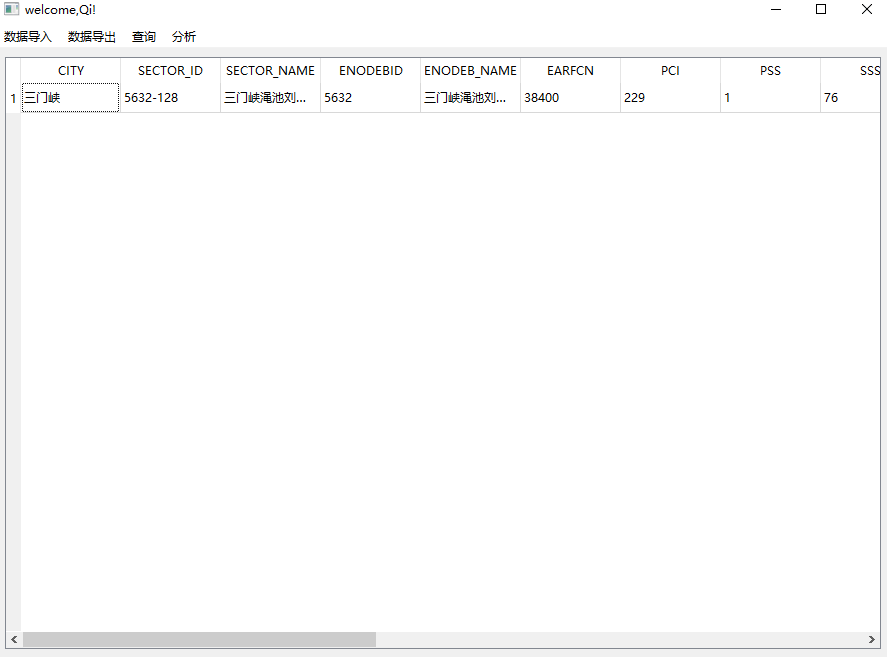


### 4.3.1小区配置信息查询

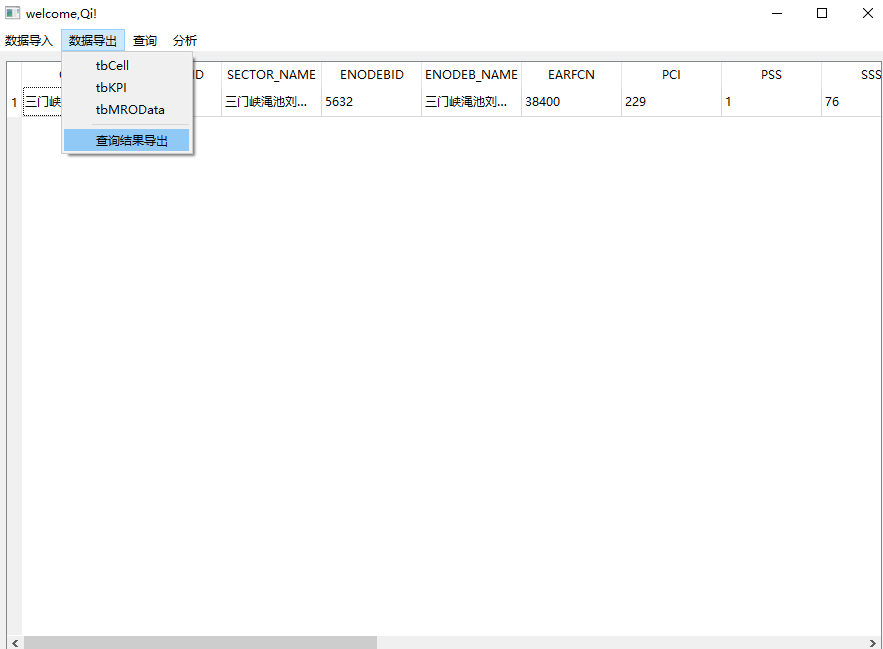
同时支持以输入框或者下拉列表的方式查询



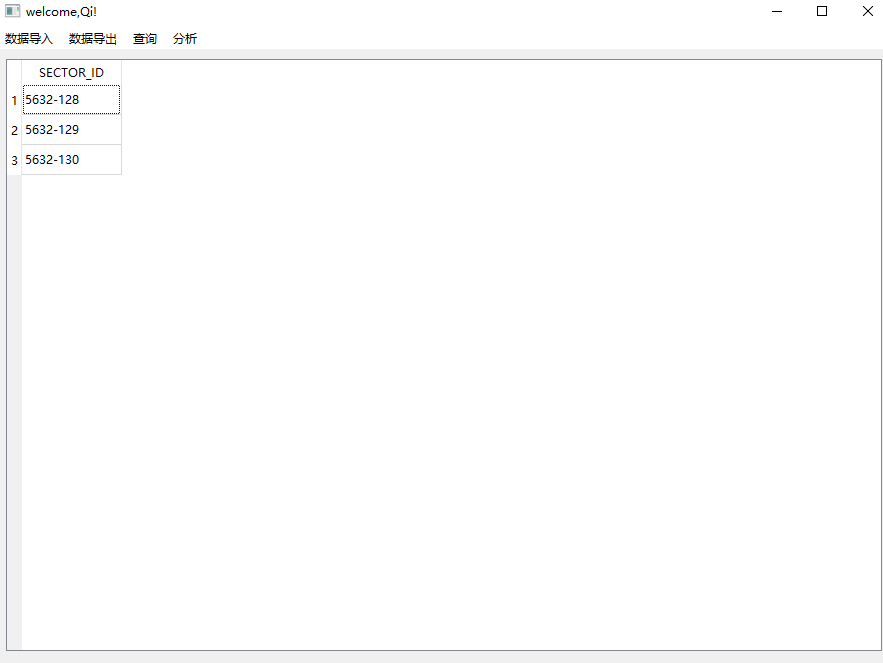
在主界面上显示查询结果：



导出查询结果：

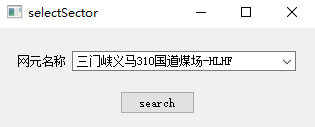


### 4.3.2 基站所属小区信息查询

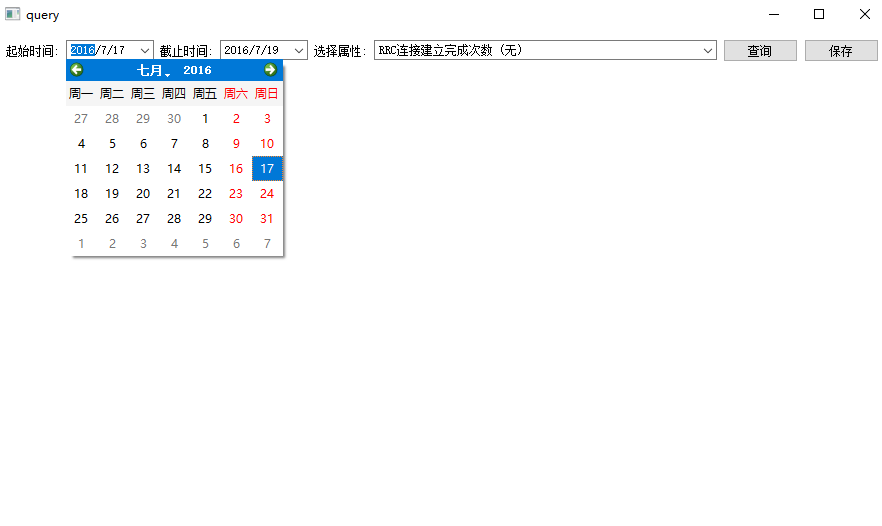


### 4.3.3 KPI指标信息查询

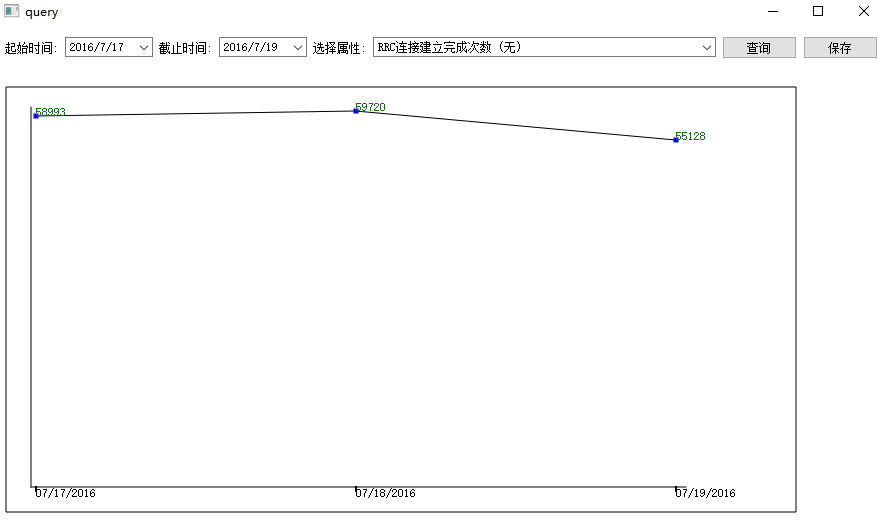
用户可在界面上根据网元名称以输入框或者下拉列表的方式选定表tbKPI中的某个特定网元：



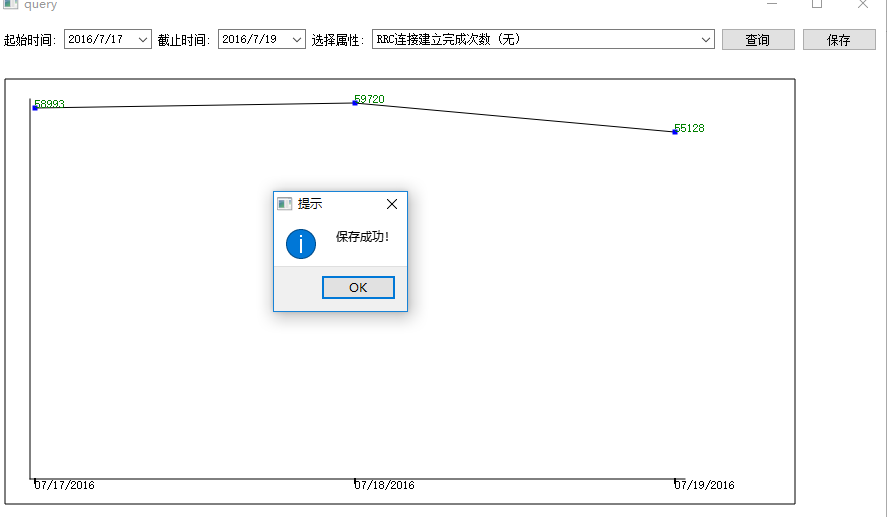
通过时间/日历控件，查询网元某个时间段（天级）某个属性值的变化情况：



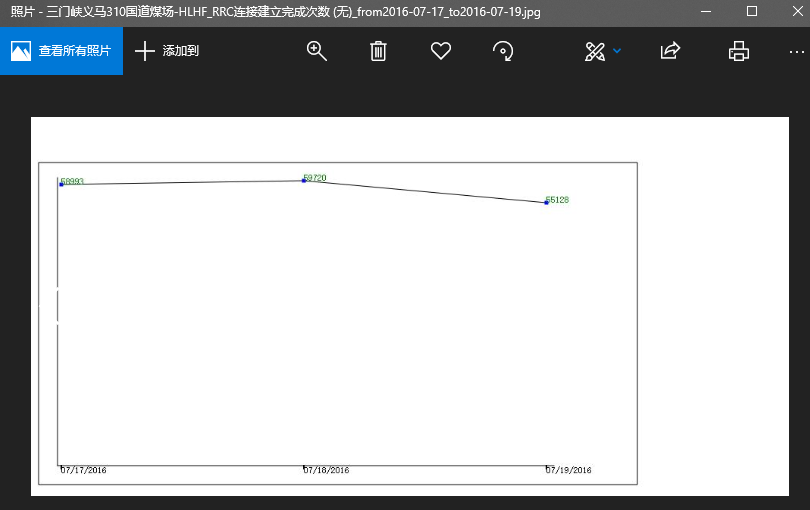
结果以折线图的方式呈现：



点击“保存”按钮，可以导出查询结果的图表：

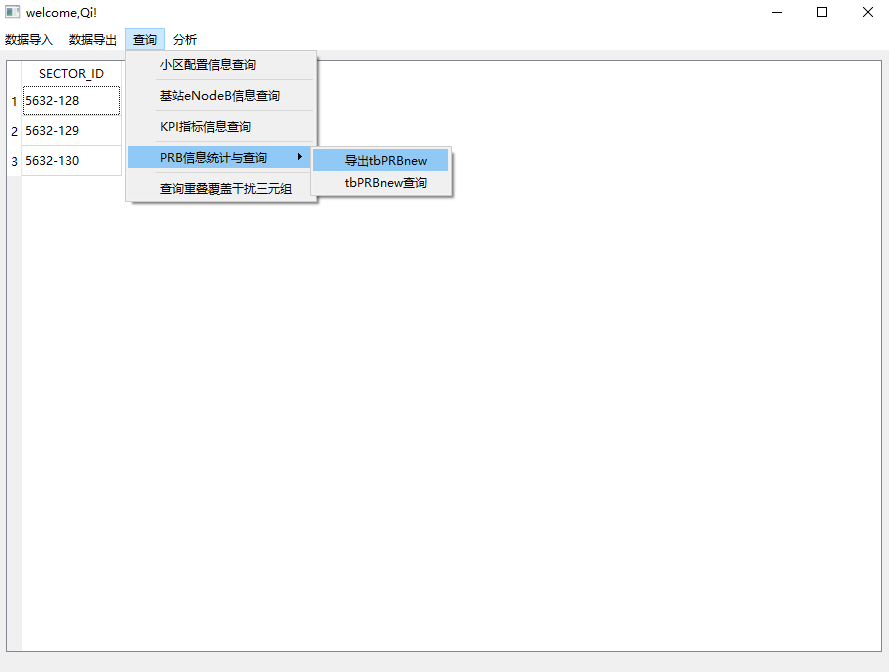


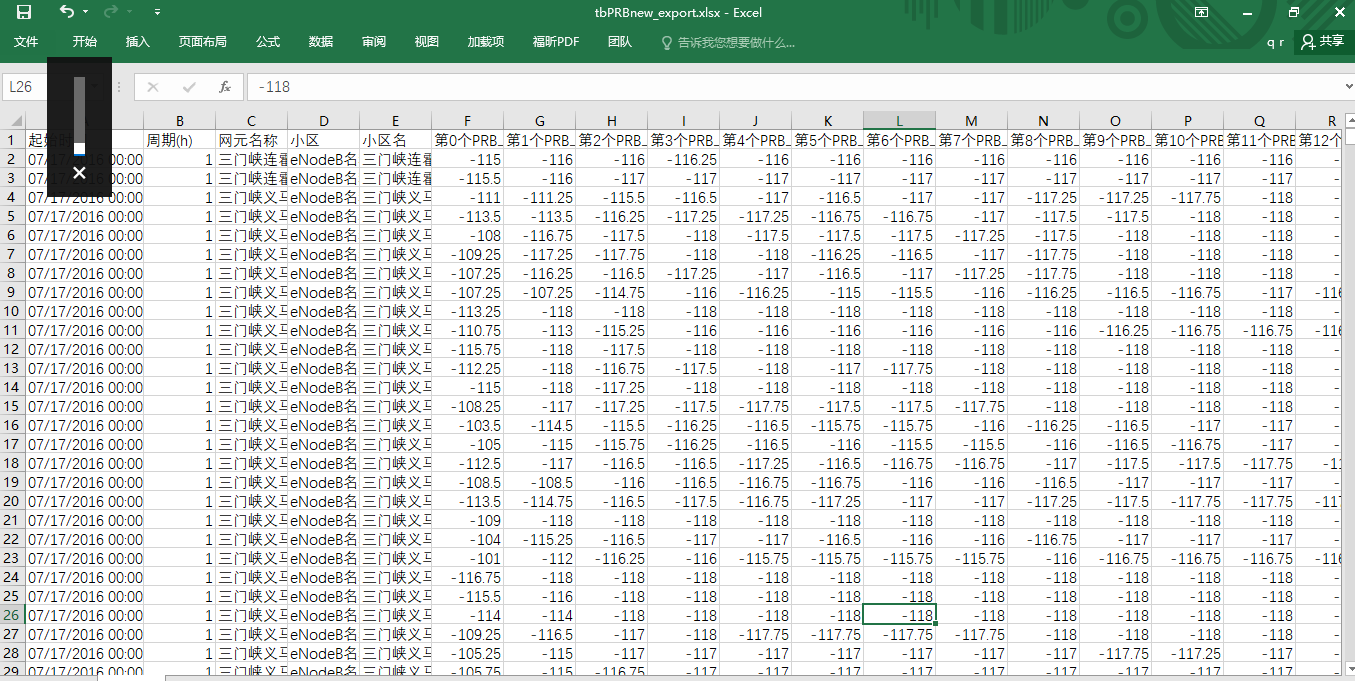
查看保存.jpg格式的图表：



### 4.3.4 PRB信息统计与查询

根据表“优化区17日-19日每PRB干扰 查询-15分钟”，统计小时级的PRB干扰数据，生成一张新表tbPRBnew，并导出到外部的excel表中：

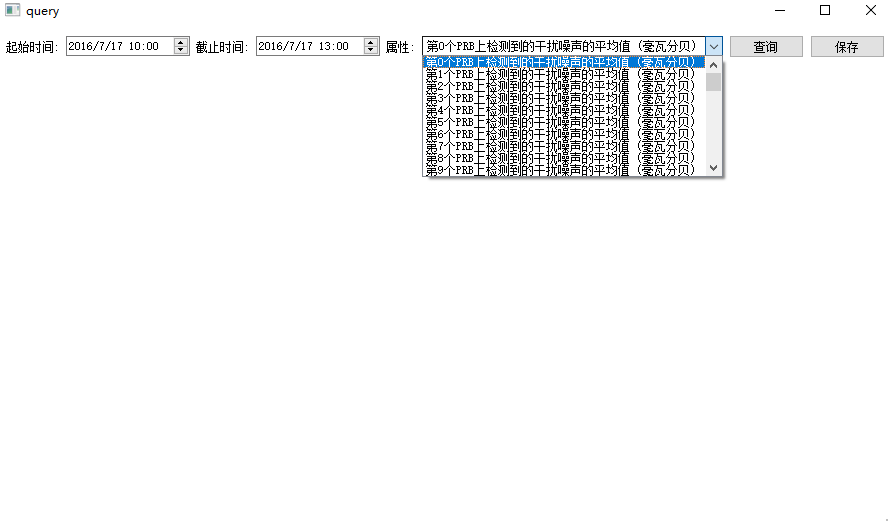


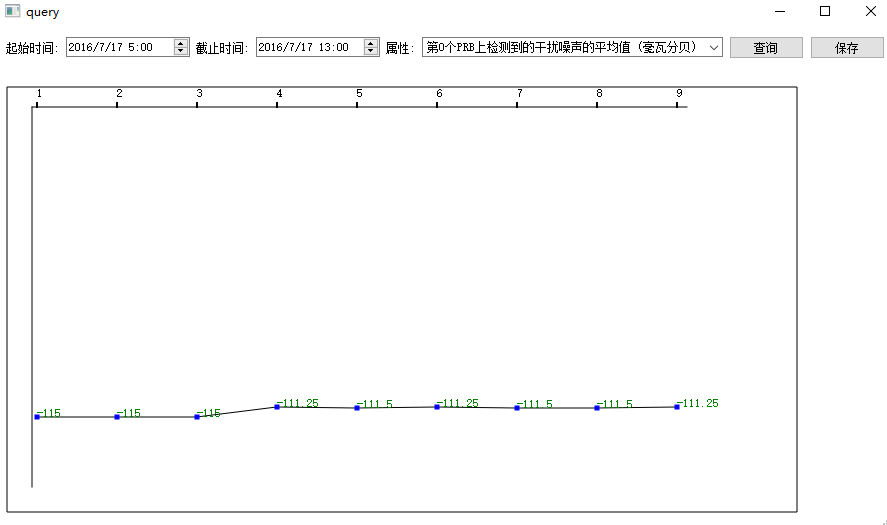


根据统计得到的小时级的PRB干扰数据表tbPRBnew，在界面上根据网元名称以输入框或者下拉列表的方式选定某个特定网元：

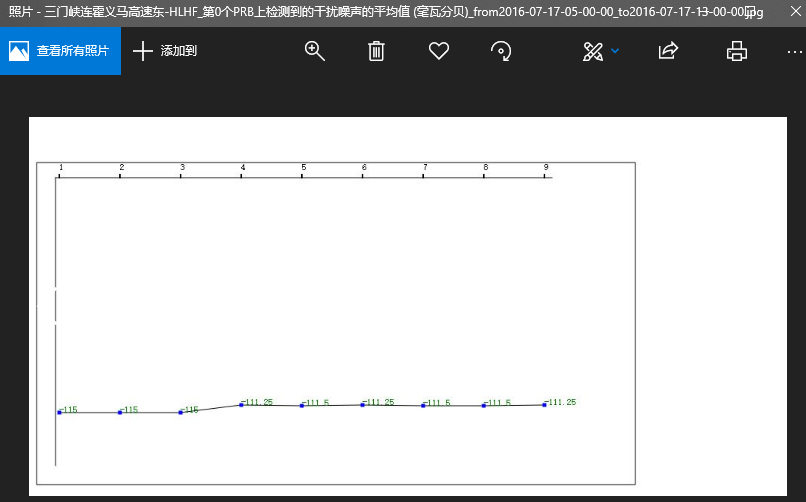


添加时间控件，查询网元某个时间段（小时级）某个属性值的变化情况，结果用折线图的方式呈现出来：





点击“保存”，可导出查询的图表结果：



### 4.3.5 重叠覆盖干扰三元组查询

根据表tbC2Inew，找出所有的小区三元组<a,b,c>（其中a,b,c互为邻小区），生成新表tbC2I3，其中有三个属性，分别是三个小区的小区ID。

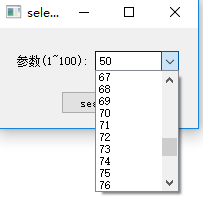
使得：

1. PrbABS6(a,b)或PrbABS6(b,a)>=x%;
2. PrbABS6(a,c)或PrbABS6(c,a)>=x%;
3. PrbABS6(b,c)或PrbABS6(c,b)>=x%;

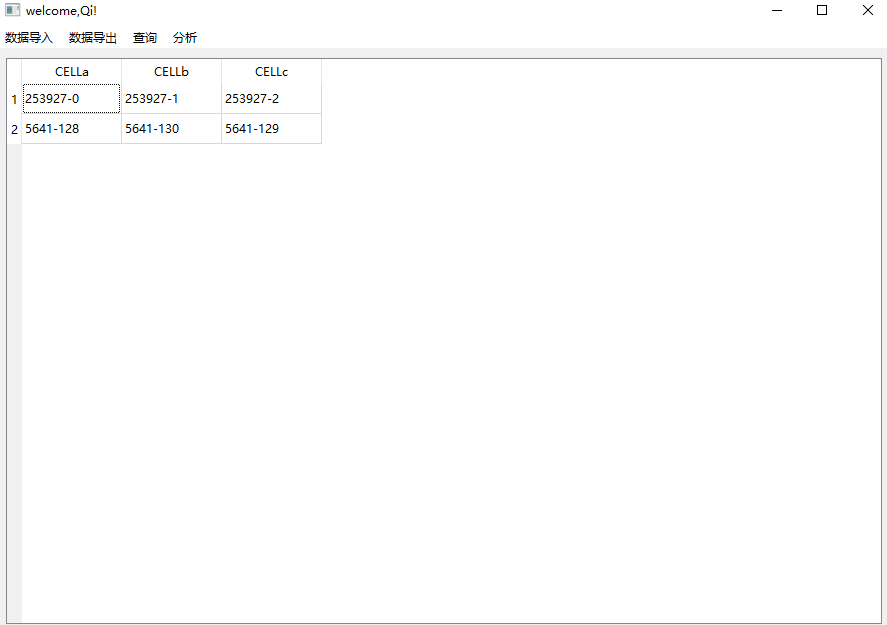
注意：PrbABS6即主邻小区RSRP差值的绝对值小于6的概率。

在根据上述方法计算出的三元组<a, b, c>中，小区a、b、c相互间信号差别不大，没有主导小区，出现严重重叠覆盖，会严重影响通话质量、下载速率等KPI性能指标。

x参数可配，在界面上输入不同的值显示不同的结果：

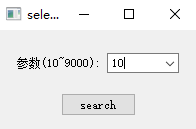


查询到的三元组显示在主界面：

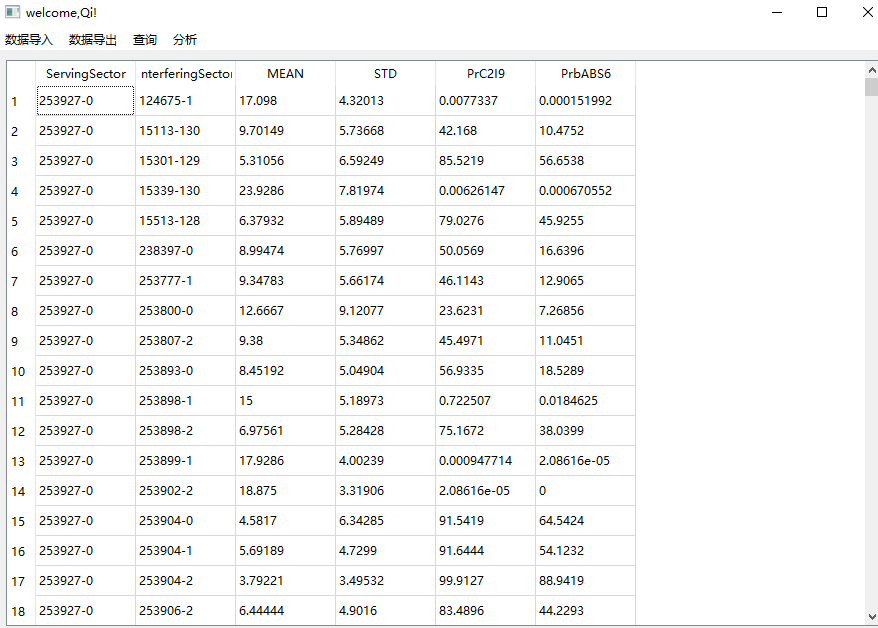


## 4.4 主邻小区C2I干扰分析

可自定义一个参数，将<LteScRSRP, LteNcRSRP>RSRP测量值对条数小于这个参数的内容先筛选掉：



根据查询条件到得到的查询结果保存在tbC2Inew，显示在主界面：



# 五、总结

祁瑞：

本次课程设计中遇到的主要问题是c++不支持从内存bulk insert或bulk copy将数据批量导入sql server数据库，而sql语句仅支持从磁盘文件将数据批量导入sql server数据库，本实验中采用的批量导入方式是QSqlQuery的execBatch()方法以实现数据批量导入sql server数据库。此外，在数据导入过程中会出现因长时间读写文件出现界面卡死的情况，在读文件的循环中添加QCoreApplication::processEvents()语句有效解决了这一问题。

通过本次小学期的数据库课程设计，我对数据库应用系统的开发周期模型有了亲身的实践经历，对数据库应用系统的设计有了更深的理解，进一步掌握了sql server数据库存储过程和函数的定义和使用，掌握了qt访问sql server数据库的API接口的使用，学会了如何用应用程序的方式实现数据的批量导入与导出，同时掌握了Qt快速读写excel文件的方法。在解决问题的过程中也提高了我分析解决问题的能力。

张嘉轩：

通过一学期的数据库课程的学习，只是掌握了数据库的理论知识，但是从理论到实践又是一个很大的挑战，在这两星期的课程设计中，亲身的感受了一下设计一个完整的数据库系统的全过程。在完成实验报告管理系统的数据库设计之后，感觉自己对数据库设计不再是局限于数据库的创建和表的创建，而是要从实际出发，结合用户的需求来设计数据库，并且对约束的选择，主键、索引的选择也是一个繁琐的过程。总之，通过此次课程设计，我感到获益匪浅。