《数据查询与修改》  
实验报告



|  |  |
| --- | --- |
| **学院：** | **计算机学院（国家示范性软件学院）** |
| **班级：** | **2019211308 2019211308 2019211308** |
| **姓名：** | **顾天阳 曾世茂 庞仕泽** |
| **学号：** | **2019211539 2019211532 2019211509** |

目录

[一、实验环境 3](#_Toc6023)

[二、 实验内容 3](#_Toc5429)

[1、 单表查询 3](#_Toc22896)

[ 查询1 3](#_Toc2461)

[ 查询2 4](#_Toc9321)

[ 查询3 5](#_Toc11329)

[  查询4 6](#_Toc15727)

[ 查询5 6](#_Toc15217)

[ 查询 6 7](#_Toc27068)

[2、 多表查询 9](#_Toc28441)

[ 查询 7 10](#_Toc21173)

[ 查询8 10](#_Toc8051)

[ 查询9 11](#_Toc6774)

[ 查询10 11](#_Toc28135)

[3、 统计查询 12](#_Toc11666)

[ 查询11 12](#_Toc27205)

[ 查询12 14](#_Toc1025)

[  查询13 14](#_Toc16003)

[4、 嵌套查询 15](#_Toc21073)

[ 查询14 15](#_Toc32153)

[ 查询15-1 17](#_Toc30105)

[ 查询15-2 19](#_Toc22083)

[ 查询16-1 22](#_Toc30650)

[ 查询16-2 23](#_Toc12449)

[ 查询17-1 24](#_Toc14069)

[ 查询17-2 26](#_Toc13843)

[ 查询18 27](#_Toc21212)

[ 查询19 28](#_Toc31942)

[5、 with临时视图查询 30](#_Toc21033)

[ 查询20 30](#_Toc7368)

[ 查询21 32](#_Toc20489)

[6、 键/函数依赖分析 34](#_Toc25245)

[ 查询22 34](#_Toc13515)

[ 查询23 35](#_Toc13233)

[7、 表的插入、删除、更新 36](#_Toc23841)

[ 查询24 36](#_Toc6935)

[ 查询25 37](#_Toc27732)

[ 查询26 37](#_Toc26991)

[ 查询27 38](#_Toc10516)

[ 查询28 39](#_Toc26872)

[三、遇到问题及解决 40](#_Toc458)

[1、版本问题 40](#_Toc28423)

[2、delete测试 41](#_Toc31906)

[3、not unique问题 41](#_Toc7818)

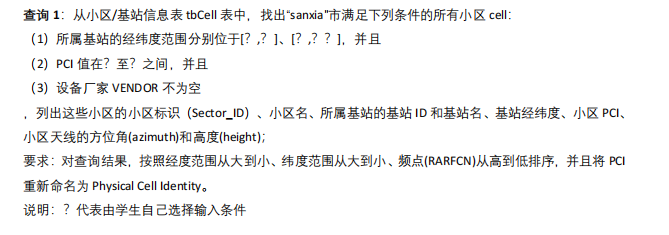
一、实验环境

本次 GaussDB 数据查询与修改实验基于 GaussDB(for openGauss) 1.4.10 进行，本地操作系统环境为 Windows 11 Home。

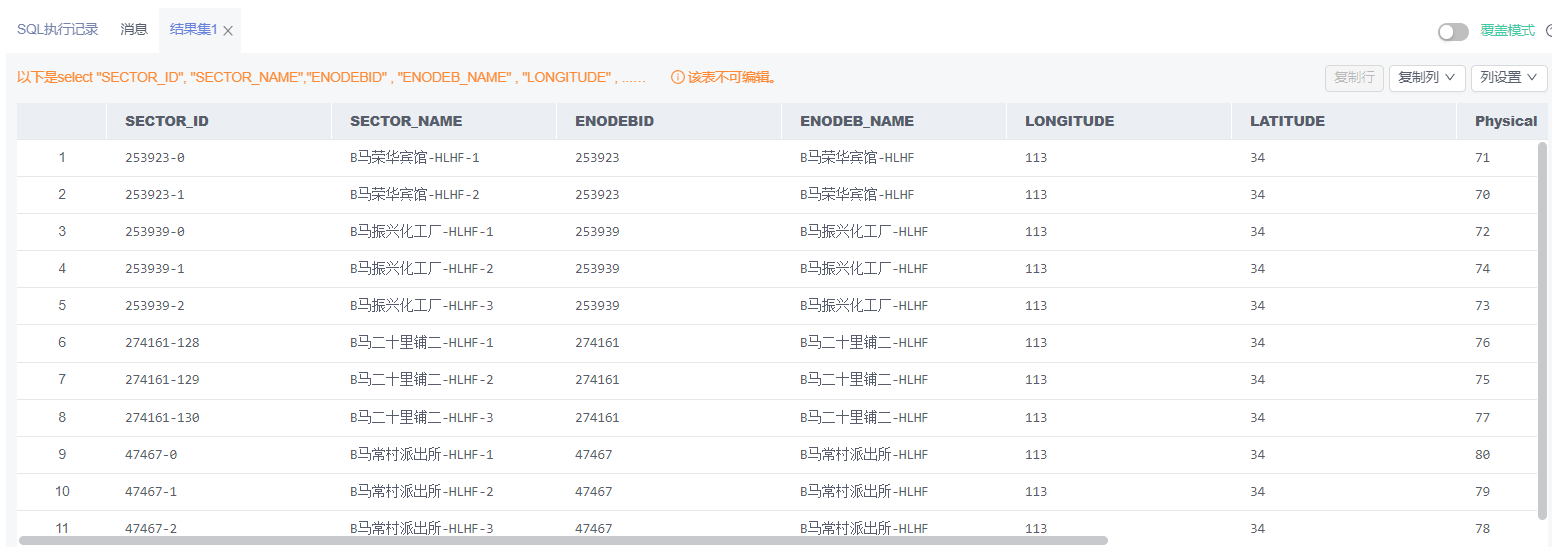
1. 实验内容
2. 单表查询

* 查询1

1. 查询内容

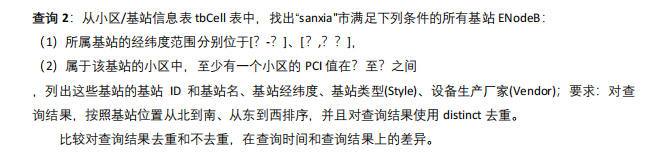


1. 查询语句
2. **select** "SECTOR\_ID", "SECTOR\_NAME","ENODEBID" , "ENODEB\_NAME" , "LONGITUDE" , "LATITUDE","PCI" **AS** "Physical Cell Identity","AZIMUTH","HEIGHT" **from** tbcell **where**
3. "CITY" = 'sanxia' AND
4. "LONGITUDE" >= 112 AND
5. "LONGITUDE" <= 114 AND
6. "PCI" >= 70 AND
7. "PCI" <= 80 AND
8. "LATITUDE" >= 34 AND
9. "LATITUDE" <= 35 AND
10. "VENDOR" NOTNULL
11. **ORDER** **BY** "LONGITUDE","LATITUDE","EARFCN" **DESC**
12. 查询结果



* 查询2

1. 查询内容



1. 查询
   1. 去重
2. **Select** **DISTINCT**  "ENODEBID" ,"ENODEB\_NAME","LONGITUDE","LATITUDE","STYLE","VENDOR"
3. **From**(**select** \*
4. **From** tbcell\_2
5. **Where** "PCI" between 40 and 50)





* 1. 不去重

1. **Select**  "ENODEBID" ,"ENODEB\_NAME","LONGITUDE","LATITUDE","STYLE","VENDOR"
2. **From**(**select** \*
3. **From** tbcell\_2
4. **Where** "PCI" between 40 and 50)

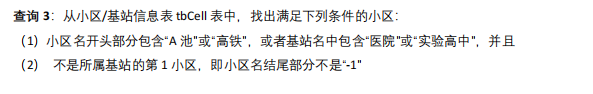




结果：添加去重后消耗时间会大幅增加。

* 查询3

1. 查询内容

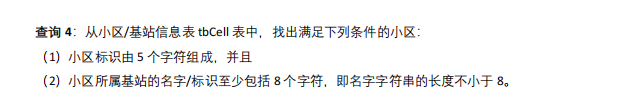


1. 查询语句
2. **SELECT** \*
3. **FROM** tbcell\_2   **WHERE**
4. ("SECTOR\_NAME" LIKE 'A池%'   OR
5. "SECTOR\_NAME" LIKE '高铁%' OR
6. "ENODEB\_NAME" LIKE '%医院%' OR
7. "ENODEB\_NAME" LIKE '%实验高中%') AND
8. "SECTOR\_NAME" NOT LIKE '%-1'
9. 查询结果



* 查询4

(1) 查询内容



(2) 查询语句

1. **SELECT** \* **FROM** tbcell\_2 **WHERE**
2. LENGTH("SECTOR\_ID") = 5 AND
3. LENGTH("ENODEB\_NAME") >= 8

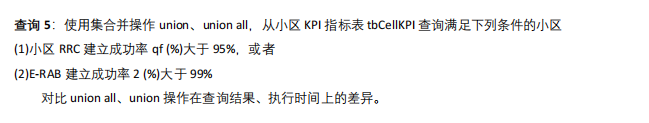
(3) 查询结果



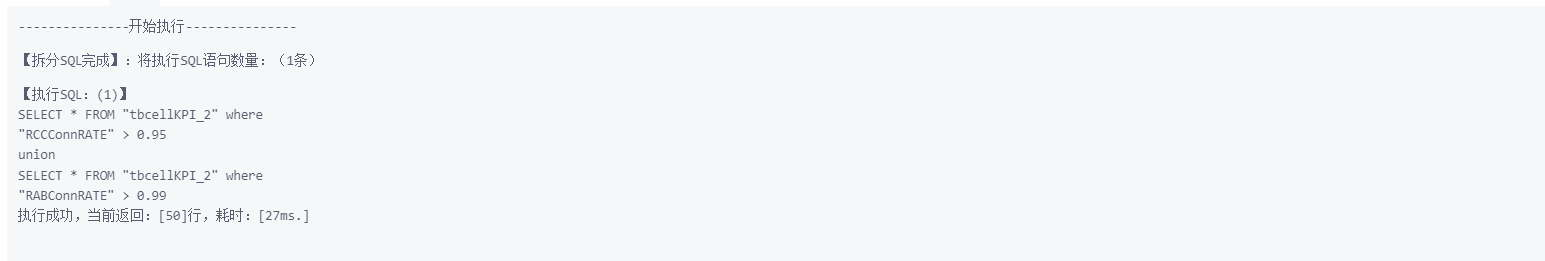
没有一条数据符合该要求

* 查询5

1. 查询内容



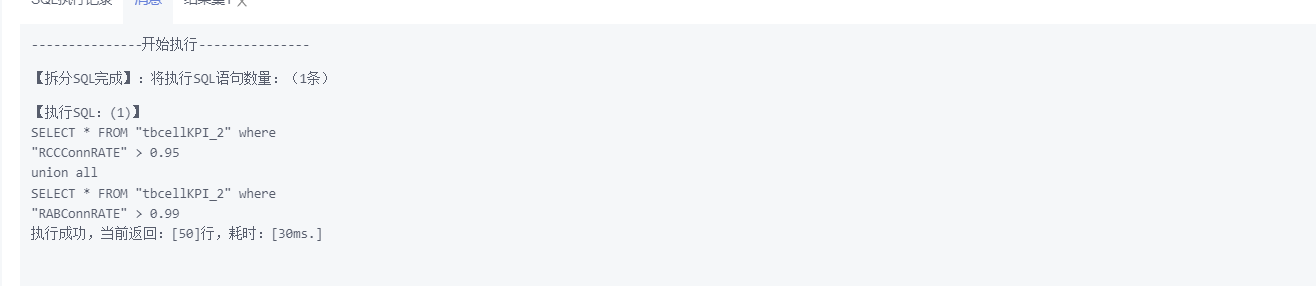
1. 查询
   1. Union
2. **SELECT** \* **FROM** "tbcellKPI\_2" **where**
3. "RCCConnRATE" > 0.95
4. **union**
5. **SELECT** \* **FROM** "tbcellKPI\_2" **where**
6. "RABConnRATE" > 0.99





* 1. Union all

1. **SELECT** \* **FROM** "tbcellKPI\_2" **where**
2. "RCCConnRATE" > 0.95
3. **union** all
4. **SELECT** \* **FROM** "tbcellKPI\_2" **where**
5. "RABConnRATE" > 0.99

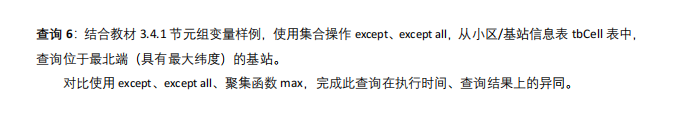




结果： Union会执行去重操作，所以Union all中会出现重复的数据。而在这个例子中，两者的运行时间没有明显差异

* 查询 6

(1) 查询内容



(2) 查询

1. Except

查询语句

1. (**select** "ENODEBID", "ENODEB\_NAME"
2. **from** tbcell\_2)
3. **except**
4. (**select** "ENODEBID", "ENODEB\_NAME"
5. **from** tbcell\_2 **as** t1

**where** exists (**select** \* **from** tbcell\_2 **as** t2 **where** t1."LATITUDE" < t2."LATITUDE"));

查询结果



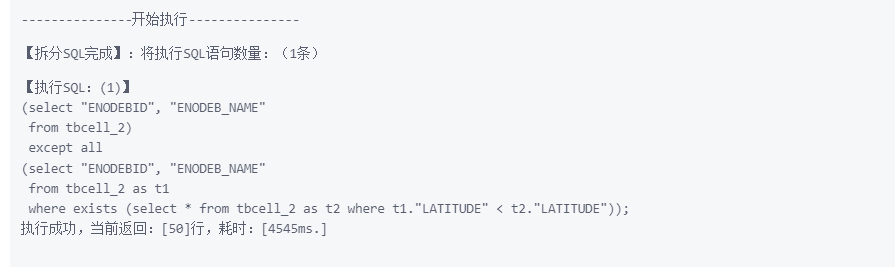


1. Except all

查询语句

1. (**select** "ENODEBID", "ENODEB\_NAME"
2. **from** tbcell\_2)
3. **except** all
4. (**select** "ENODEBID", "ENODEB\_NAME"
5. **from** tbcell\_2 **as** t1
6. **where** exists (**select** \* **from** tbcell\_2 **as** t2 **where** t1."LATITUDE" < t2."LATITUDE"));

查询结果



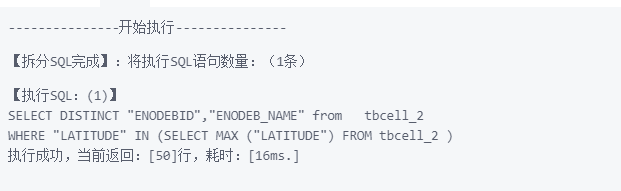


1. Max

 查询语句

1. **SELECT distinct** "ENODEBID", "ENODEB\_NAME" **FROM** tbcell\_2
2. **WHERE** "LATITUDE" IN(**SELECT** **MAX**("LATITUDE") **FROM** tbcell\_2)

查询结果



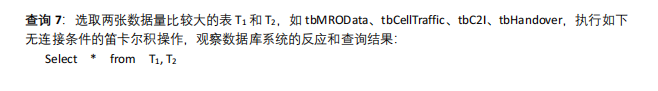


 结果分析：直接使用max函数的速度最快，远超使用except或except all。而直接使用max函数且使用distinct后结果无重复值，使用except all可能存在重复值，except也不会存在重复值

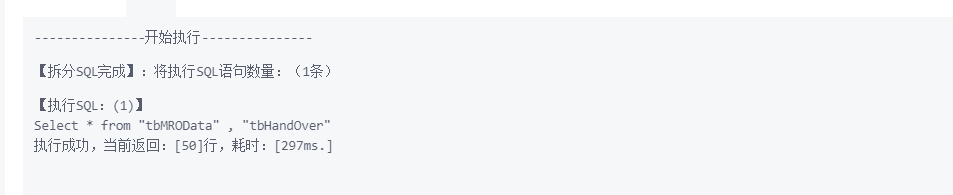
1. 多表查询

* 查询 7

1. 查询内容



1. 查询语句
2. **Select** \* **from** "tbMROData" , "tbHandOver"
3. 查询结果

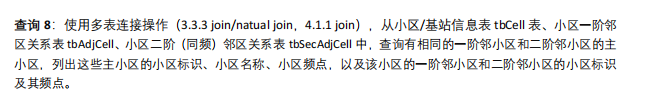




结果分析：两表连接的速度较慢。 连接结果两个表的全连接

* 查询8

1. 查询内容



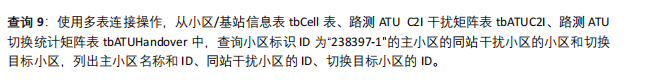
1. 查询语句
2. **SELECT** "tbAdjCell"."S\_SECTOR\_ID", "tbcell\_2"."SECTOR\_NAME", "tbAdjCell"."S\_EARFCN", "tbAdjCell"."N\_SECTOR\_ID" **AS** "ADJ\_CELL", "tbSecAdjCell"."N\_SECTOR\_ID" **AS** "SEC\_ADJ"
3. , "tbAdjCell"."N\_EARFCN"
4. **FROM** "tbAdjCell"
5. JOIN "tbSecAdjCell" **ON** "tbAdjCell"."S\_SECTOR\_ID" = "tbSecAdjCell"."S\_SECTOR\_ID", "tbcell\_2"
6. **WHERE** "ADJ\_CELL" = "SEC\_ADJ"
7. AND "tbAdjCell"."S\_SECTOR\_ID" = "tbcell\_2"."SECTOR\_ID";
8. 查询结果





* 查询9

(1) 查询内容



(2) 查询语句

1. **SELECT**  "tbcell\_2" ."SECTOR\_ID" , "tbATUC2I" ."NCELL\_ID" , "tbHandOver" ."NCELL"
2. **from** "tbcell\_2"  join "tbATUC2I" **on** "tbcell\_2" ."SECTOR\_ID" = "tbATUC2I" ."SECTOR\_ID" JOIN "tbHandOver" **ON** "tbcell\_2" ."SECTOR\_ID" = "tbHandOver" ."SCELL"
3. **WHERE** "tbcell\_2" ."SECTOR\_ID" = '238397-1'

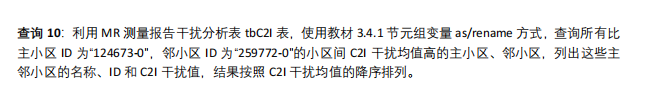
(3) 查询结果



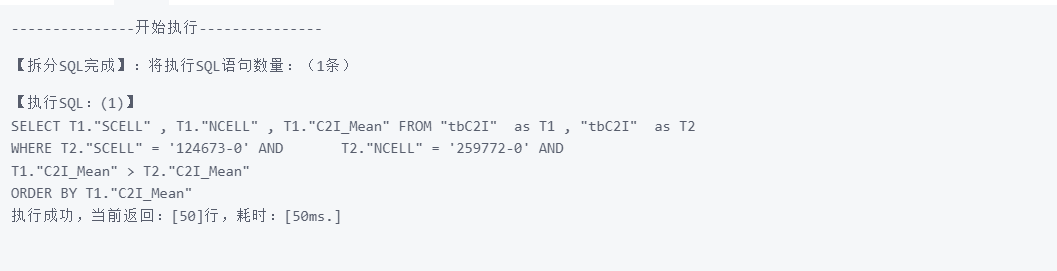


* 查询10

1. 查询内容



1. 查询语句
2. **SELECT** T1."SCELL" , T1."NCELL" , T1."C2I\_Mean" **FROM** "tbC2I"  **as** T1 , "tbC2I"  **as** T2
3. **WHERE** T2."SCELL" = '124673-0' AND   T2."NCELL" = '259772-0' AND
4. T1."C2I\_Mean" > T2."C2I\_Mean"
5. **ORDER** **BY** T1."C2I\_Mean"
6. 查询结果

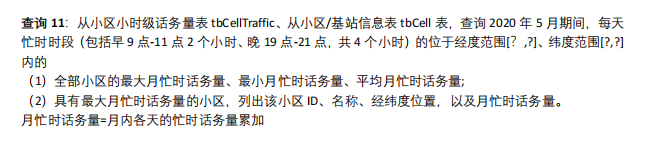




1. 统计查询

* 查询11

(1) 查询内容

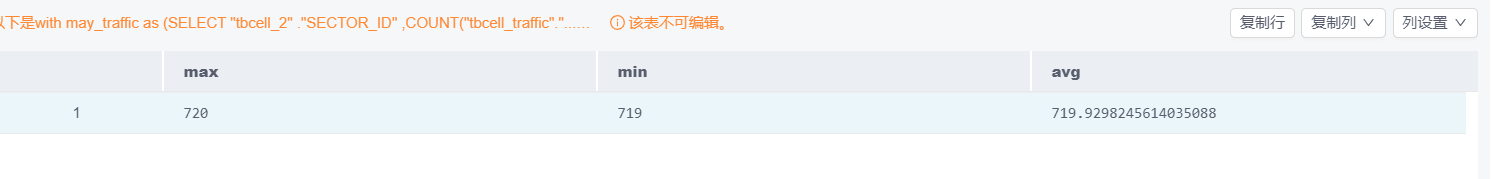


1. 全部小区的最大月忙时话务量、最小月忙时话务量、平均月忙时话务量;

 查询语句

1. **with** may\_traffic **as**
2. (
3. **SELECT**  "tbcell\_2" ."SECTOR\_ID" ,COUNT("tbcell\_traffic"."celltraffic") **as** "traffic\_May" **FROM** "tbcell\_traffic"  JOIN "tbcell\_2"  **ON** "tbcell\_traffic" .sectorid  = "tbcell\_2" ."SECTOR\_ID"
4. **WHERE** "tbcell\_traffic" ."stadate" > '2020-05-01 00:00:00' and "tbcell\_traffic" ."stadate" < '2020-06-01 :00:00'
5. **GROUP** **BY** ( "tbcell\_2" ."SECTOR\_ID" )
6. )
8. **SELECT** **max**("traffic\_May") , **MIN** ("traffic\_May") , AVG ("traffic\_May") **FROM** may\_traffic

 查询结果



1. 具有最大月忙时话务量的小区，列出该小区的ID，名称，经纬度，月忙时话务量

查询语句

1. **WITH**  may\_traffic **as**(
2. **SELECT**  "tbcell\_2" ."SECTOR\_ID" ,COUNT("tbcell\_traffic"."celltraffic") **as** "traffic\_May" **FROM** "tbcell\_traffic"  JOIN "tbcell\_2"  **ON** "tbcell\_traffic" .sectorid  = "tbcell\_2" ."SECTOR\_ID"
3. **WHERE** "tbcell\_traffic" ."stadate" > '2020-05-01 00:00:00' and "tbcell\_traffic" ."stadate" < '2020-06-01 :00:00'
4. **GROUP** **BY** ( "tbcell\_2" ."SECTOR\_ID" ))
6. **SELECT** "tbcell\_2"."SECTOR\_ID" , "SECTOR\_NAME","LONGITUDE" ,"LATITUDE" , "may\_traffic"."traffic\_May"
7. **FROM**        "tbcell\_2" join   "may\_traffic" **on** "tbcell\_2"."SECTOR\_ID" = "may\_traffic"."SECTOR\_ID"
8. **WHERE**  "may\_traffic"."traffic\_May"   = (**SELECT** **max**("traffic\_May") **from** may\_traffic)

查询结果





* 查询12

1. 查询内容



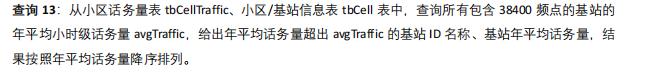
1. 查询语句
2. **SELECT** "tboptcell"."SECTOR\_ID",COUNT ("N\_SECTOR\_ID" ) **AS** ADJCOUNT  **FROM** "tbAdjCell" JOIN "tboptcell" **on** "tboptcell"."SECTOR\_ID" = "tbAdjCell"."S\_SECTOR\_ID"
3. **GROUP** **BY** "tboptcell"."SECTOR\_ID"
4. **HAVING** (ADJCOUNT>10)
5. **ORDER** **BY** ADJCOUNT **DESC**
7. 查询结果





* 查询13

(1) 查询内容



(2) 查询语句

1. **WITH** AVG\_TRAFFIC **AS**
2. (
3. **SELECT** AVG("celltraffic") **FROM**  "tbcell\_traffic" join "tbcell\_2"  **on** "tbcell\_traffic"."sectorid" = "tbcell\_2" ."SECTOR\_ID"
4. **WHERE** "tbcell\_2"."ENODEBID" IN (**SELECT** **DISTINCT** "ENODEBID" **FROM**  "tbcell\_2"  **WHERE** "EARFCN"  = 38400)
5. )
6. **SELECT**  "ENODEBID","ENODEB\_NAME",AVG("celltraffic") **FROM**     "tbcell\_traffic" join "tbcell\_2"  **on** "tbcell\_traffic"."sectorid" = "tbcell\_2" ."SECTOR\_ID"
7. **GROUP** **BY** ("ENODEBID","ENODEB\_NAME")
8. **HAVING** AVG("celltraffic") > (**SELECT** "avg" **from** AVG\_TRAFFIC)
9. **ORDER** **BY** AVG("celltraffic") **DESC**

(3) 查询结果





1. 嵌套查询

* 查询14

（1）查询内容



（2）查询语句

SELECT "SECTOR\_ID","SECTOR\_NAME","HEIGHT"

FROM "tbcell\_invariable"

WHERE "HEIGHT" > SOME

(SELECT "HEIGHT"

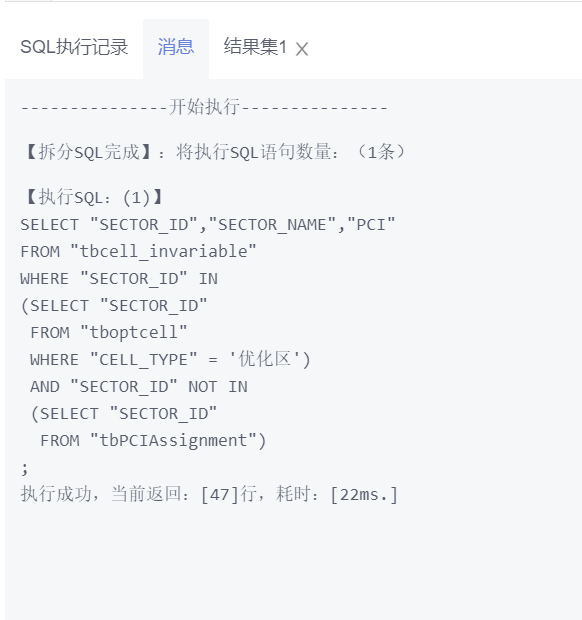
 FROM "tbcell\_invariable"

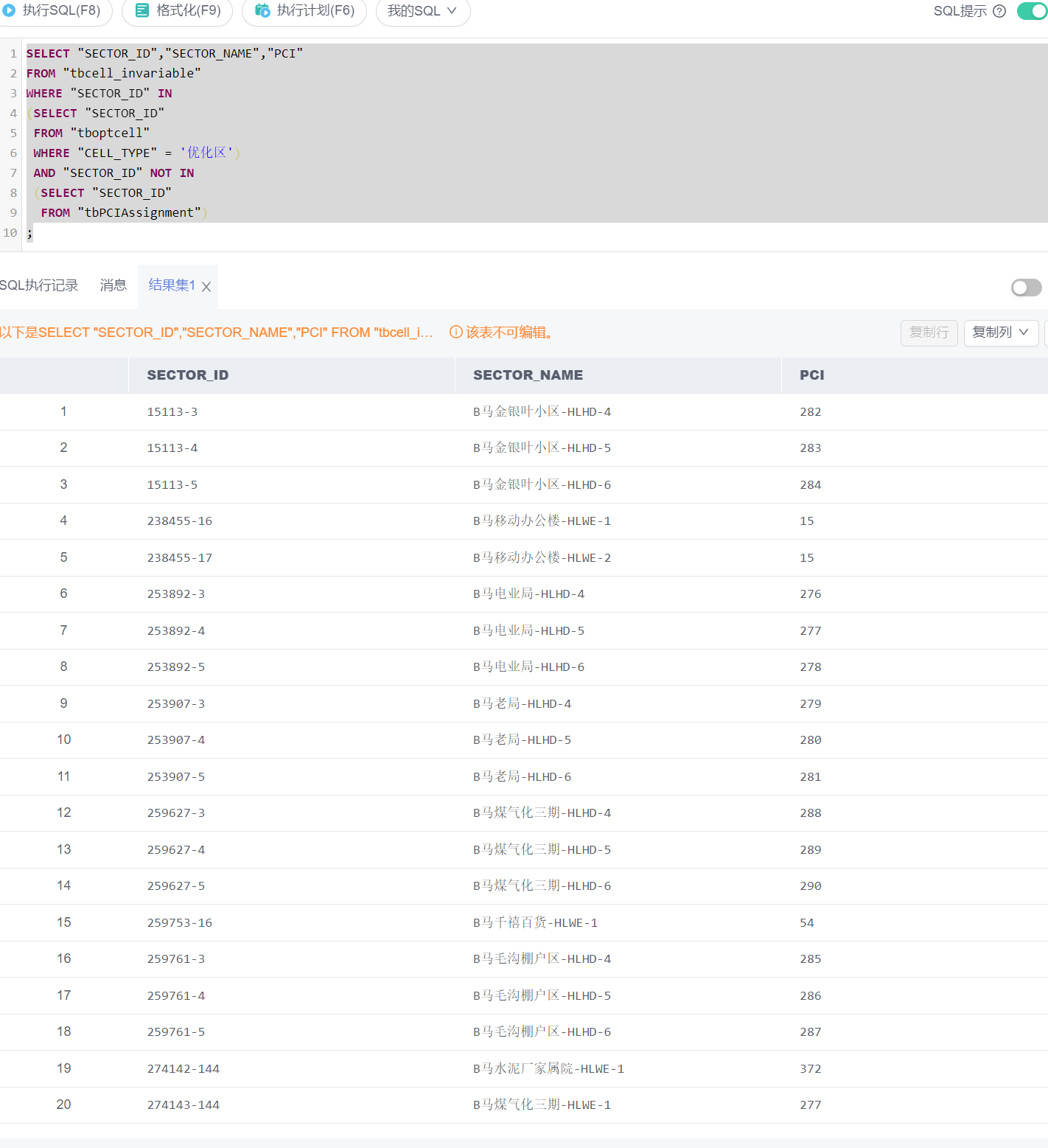
 WHERE "LONGITUDE" BETWEEN 111.5 AND 112.5

 AND "LATITUDE" BETWEEN 33.5 AND 34.5)

;

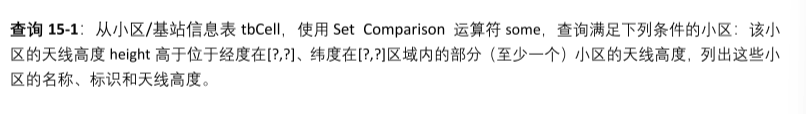
（3）查询结果





* 查询15-1

（1）查询内容



（2）查询语句

SELECT "SECTOR\_ID","SECTOR\_NAME","PCI"

SELECT "SECTOR\_ID","SECTOR\_NAME","HEIGHT"

FROM "tbcell\_invariable"

WHERE "HEIGHT" > SOME

(SELECT "HEIGHT"

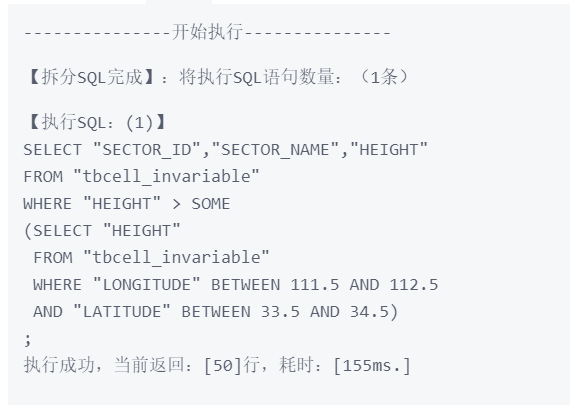
 FROM "tbcell\_invariable"

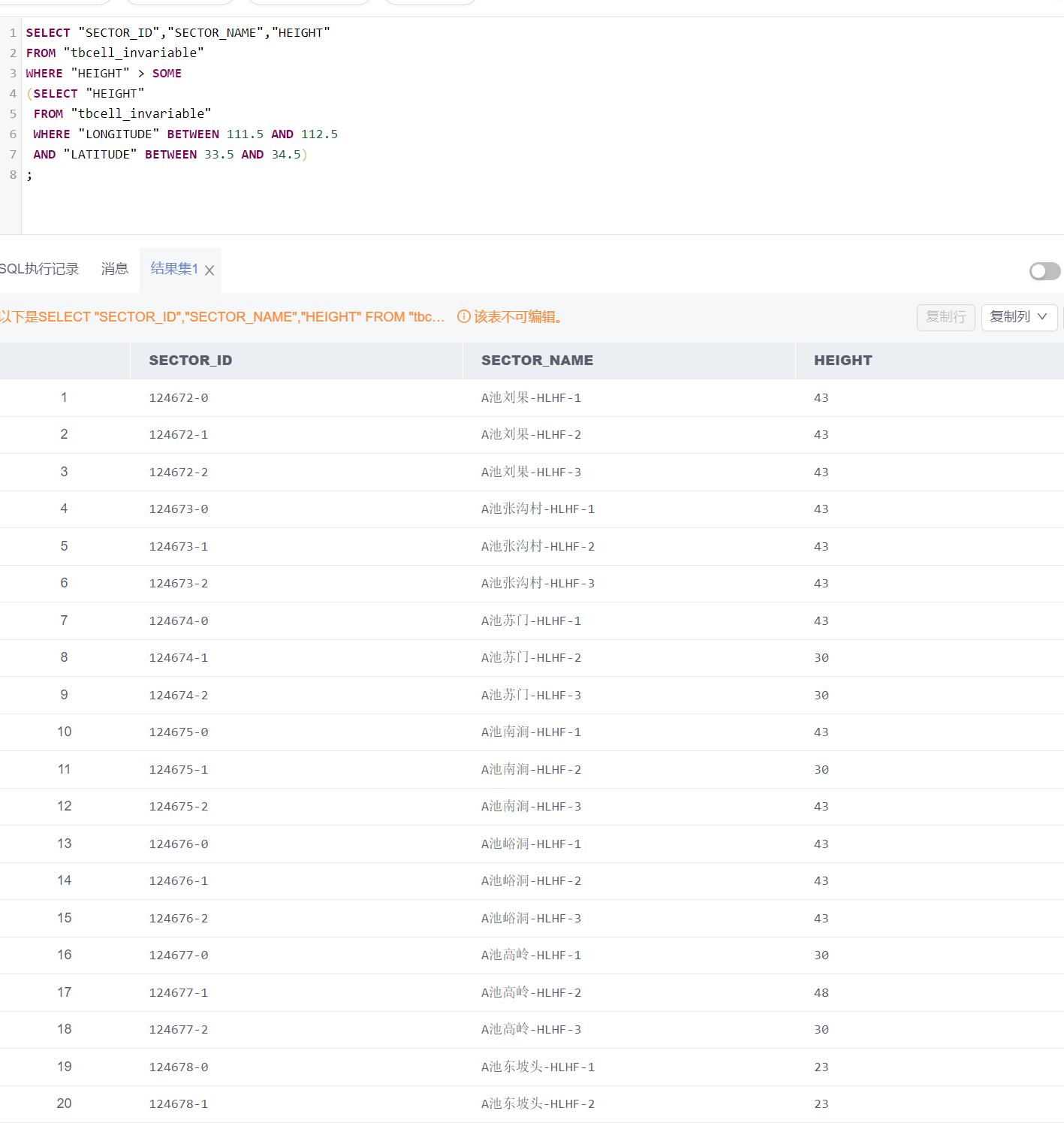
 WHERE "LONGITUDE" BETWEEN 111.5 AND 112.5

 AND "LATITUDE" BETWEEN 33.5 AND 34.5)

;

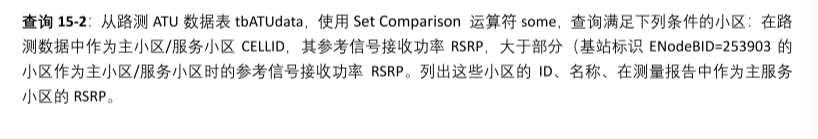
（3）查询结果





* 查询15-2

（1）查询内容



（2）查询语句

建表：tbATUdata

create table "tbATUData"

(

  "seq" BIGINT ,

  "FileName" VARCHAR (255),

  "Time" VARCHAR (100),

  "Longitude" float,

  "Latitude" float,

  "CellID" VARCHAR (50),

  "TAC" int,

  "EARFCN" int,

  "PCI" SMALLINT ,

  "RSRP" float,

  "RS\_SINR" float,

  "NCell\_ID\_1" VARCHAR (50),

  "NCell\_EARFCN\_1" int,

  "NCell\_PCI\_1" SMALLINT ,

  "NCell\_RSRP\_1" float,

  "NCell\_ID\_2" VARCHAR (50),

  "NCell\_EARFCN\_2" int,

  "NCell\_PCI\_2" SMALLINT ,

  "NCell\_RSRP\_2" float,

  "NCell\_ID\_3" VARCHAR (50),

  "NCell\_EARFCN\_3" int,

  "NCell\_PCI\_3" SMALLINT ,

  "NCell\_RSRP\_3" float,

  "NCell\_ID\_4" VARCHAR (50),

  "NCell\_EARFCN\_4" int,

  "NCell\_PCI\_4" SMALLINT ,

  "NCell\_RSRP\_4" float,

  "NCell\_ID\_5" VARCHAR (50),

  "NCell\_EARFCN\_5" int,

  "NCell\_PCI\_5" SMALLINT ,

  "NCell\_RSRP\_5" float,

  "NCell\_ID\_6" VARCHAR (50),

  "NCell\_EARFCN\_6" int,

  "NCell\_PCI\_6" SMALLINT ,

  "NCell\_RSRP\_6" float,

  primary key ("seq", "FileName")

);

查询：

SELECT A."CellID",C."SECTOR\_NAME",A."EARFCN"

FROM "tbcell\_invariable" as C,"tbATUData" as A

WHERE "RSRP" > SOME

(SELECT a."RSRP"

 FROM "tbATUData" as a,"tbcell\_invariable" as c

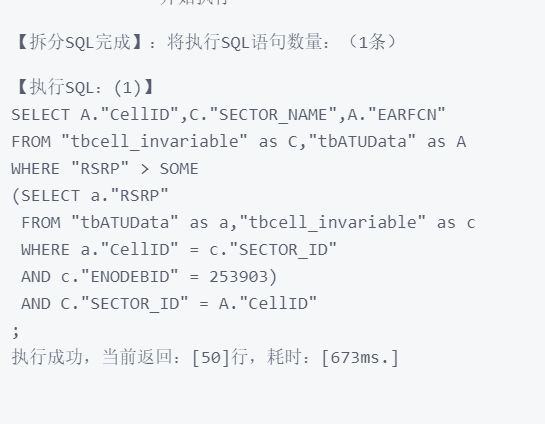
 WHERE a."CellID" = c."SECTOR\_ID"

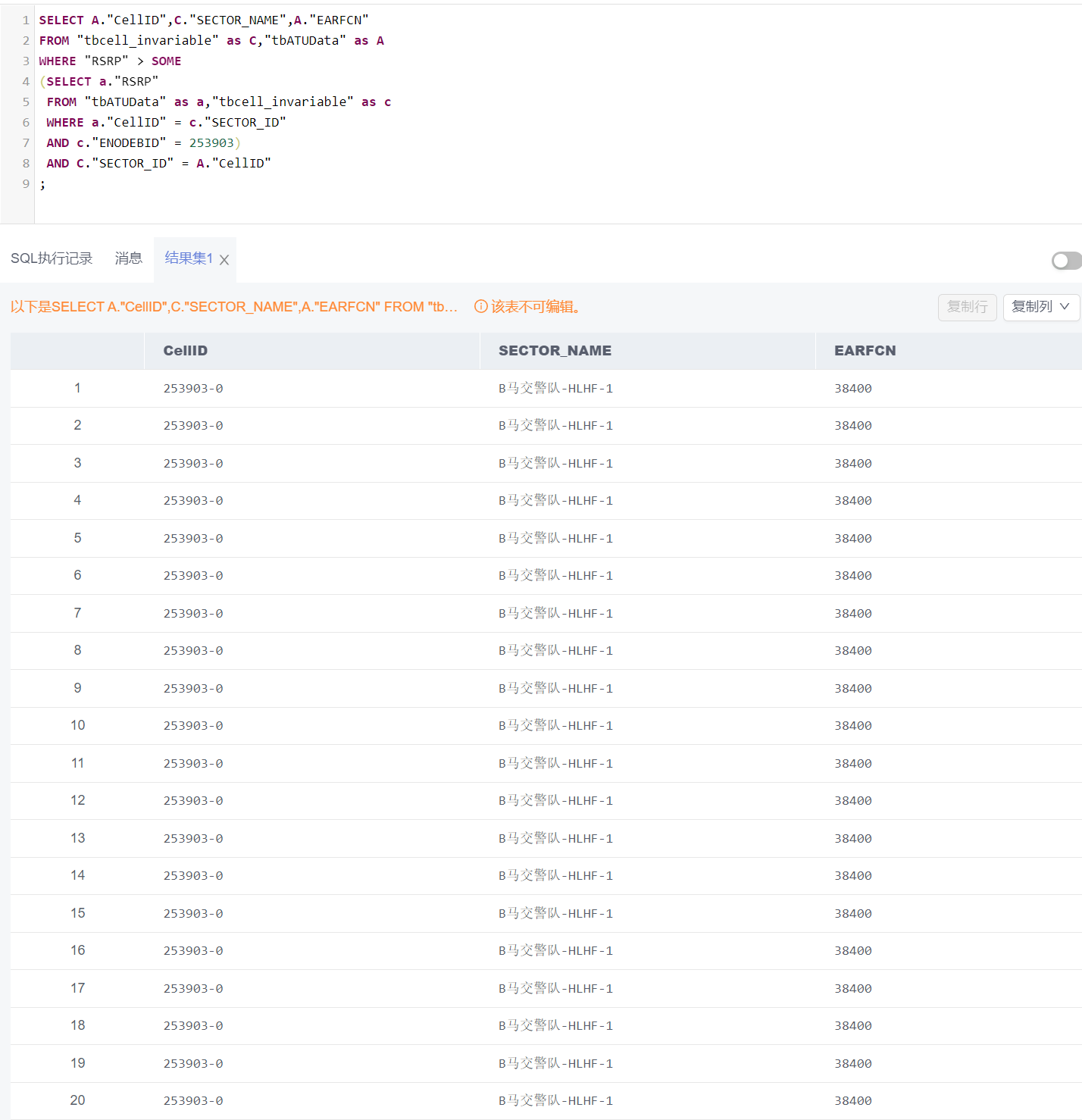
 AND c."ENODEBID" = 253903)

 AND C."SECTOR\_ID" = A."CellID"

;

（3）查询结果





* 查询16-1

（1）查询内容



（2）查询语句

SELECT "sectorid"

FROM "tbcell\_traffic"

GROUP BY "sectorid"

HAVING SUM ("celltraffic") >= ALL

(

  SELECT SUM ("celltraffic")

  FROM "tbcell\_traffic"

  GROUP BY "sectorid"

)

;

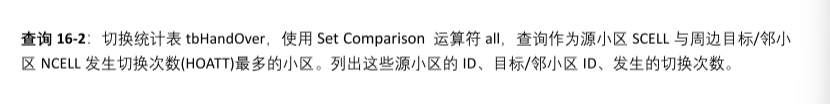
（3）查询结果





* 查询16-2

（1）查询内容



（2）查询语句

SELECT "SCELL" , "NCELL"

FROM "tbHandOver\_invariable"

WHERE "HOATT" >= ALL

(

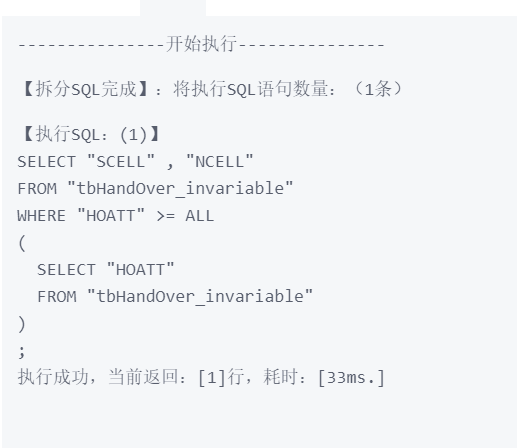
  SELECT "HOATT"

  FROM "tbHandOver\_invariable"

)

;

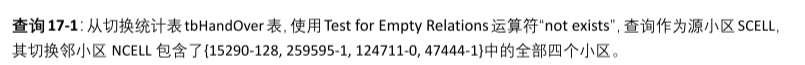
（3）查询结果





* 查询17-1

（1）查询内容



（2）查询语句

SELECT DISTINCT "SCELL"

FROM "tbHandOver\_invariable" as H

WHERE NOT EXISTS

(

  (

    SELECT "NCELL"

    FROM "tbHandOver\_invariable"

    WHERE "NCELL" = '15290-128' OR "NCELL" = '259595-1' OR "NCELL" = '124711-0' OR "NCELL" = '47444-1'

  )

  EXCEPT

  (

    SELECT I."NCELL"

    FROM "tbHandOver\_invariable" as I

    WHERE H."SCELL" = I."SCELL"

  )

)

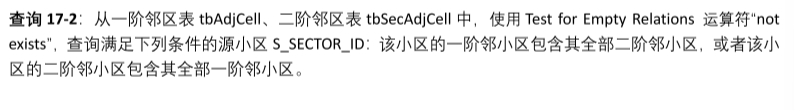
（3）查询结果





* 查询17-2

（1）查询内容



（2）查询语句

SELECT DISTINCT "S\_SECTOR\_ID"

FROM "tbAdjCell\_invariable" as A

WHERE NOT EXISTS

(

  (

    SELECT "N\_SECTOR\_ID"

    FROM "tbSecAdjCell" as B

    WHERE B."S\_SECTOR\_ID" = A."S\_SECTOR\_ID"

  )

  EXCEPT

  (

    SELECT "N\_SECTOR\_ID"

    FROM "tbAdjCell\_invariable" as C

    WHERE A."S\_SECTOR\_ID" = C."S\_SECTOR\_ID"

  )

)

OR NOT EXISTS

(

  (

    SELECT "N\_SECTOR\_ID"

    FROM "tbAdjCell\_invariable" as D

    WHERE D."S\_SECTOR\_ID" = A."S\_SECTOR\_ID"

  )

  EXCEPT

  (

    SELECT "N\_SECTOR\_ID"

    FROM "tbSecAdjCell" as E

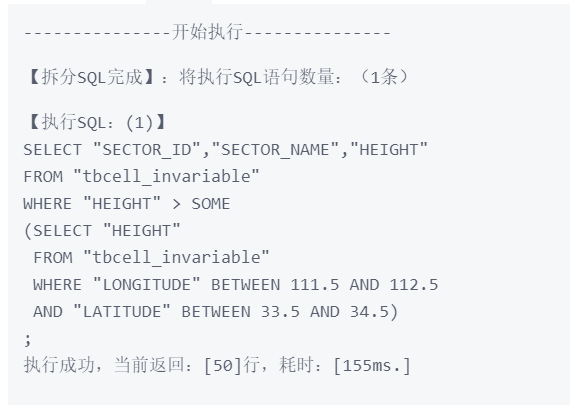
    WHERE A."S\_SECTOR\_ID" = E."S\_SECTOR\_ID"

  )

)

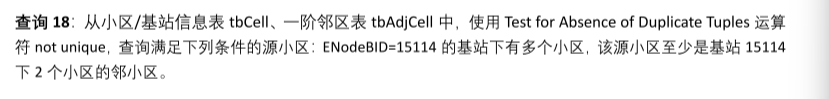
;

（3）查询结果



* 查询18

（1）查询内容



（2）查询语句

SELECT "SECTOR\_ID"

FROM "tbcell\_invariable" as A

WHERE NOT UNIQUE

(

  SELECT B."SECTOR\_ID"

  FROM "tbcell\_invariable" as B, "tbAdjCell\_invariable" as C

  WHERE C."ENODEBID" = 15114 AND

  B."SECTOR\_ID" = C."N\_SECTOR\_ID" AND

  A."SECTOR\_ID" = B."SECTOR\_ID"

)

;

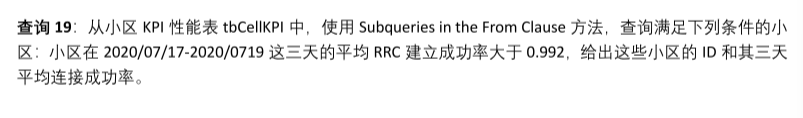
（3）查询结果

**postgre sql没有not unique**



* 查询19

（1）查询内容



（2）查询语句

SELECT "SECTOR\_ID",avg\_RCCRATE

FROM

(

  SELECT "SECTOR\_ID",AVG ("RCCConnRATE") as avg\_RCCRATE

  FROM "tbcell\_invariable" as A, "tbcellKPI\_2" as B

  WHERE A."SECTOR\_NAME" = B."SECTOR\_NAME"

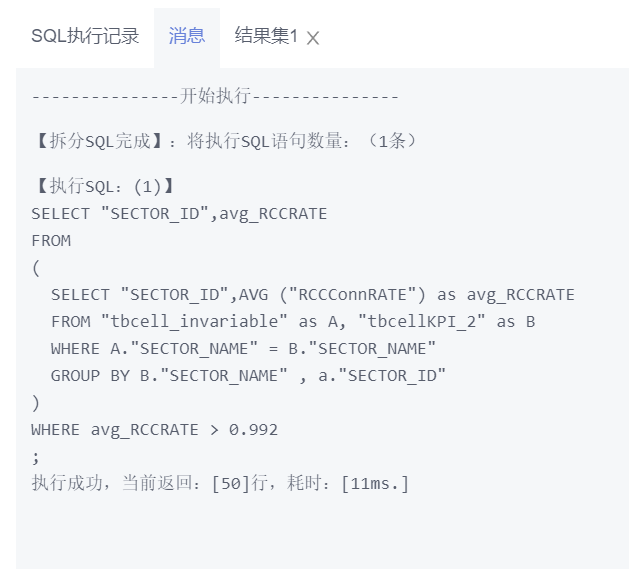
  GROUP BY B."SECTOR\_NAME" , a."SECTOR\_ID"

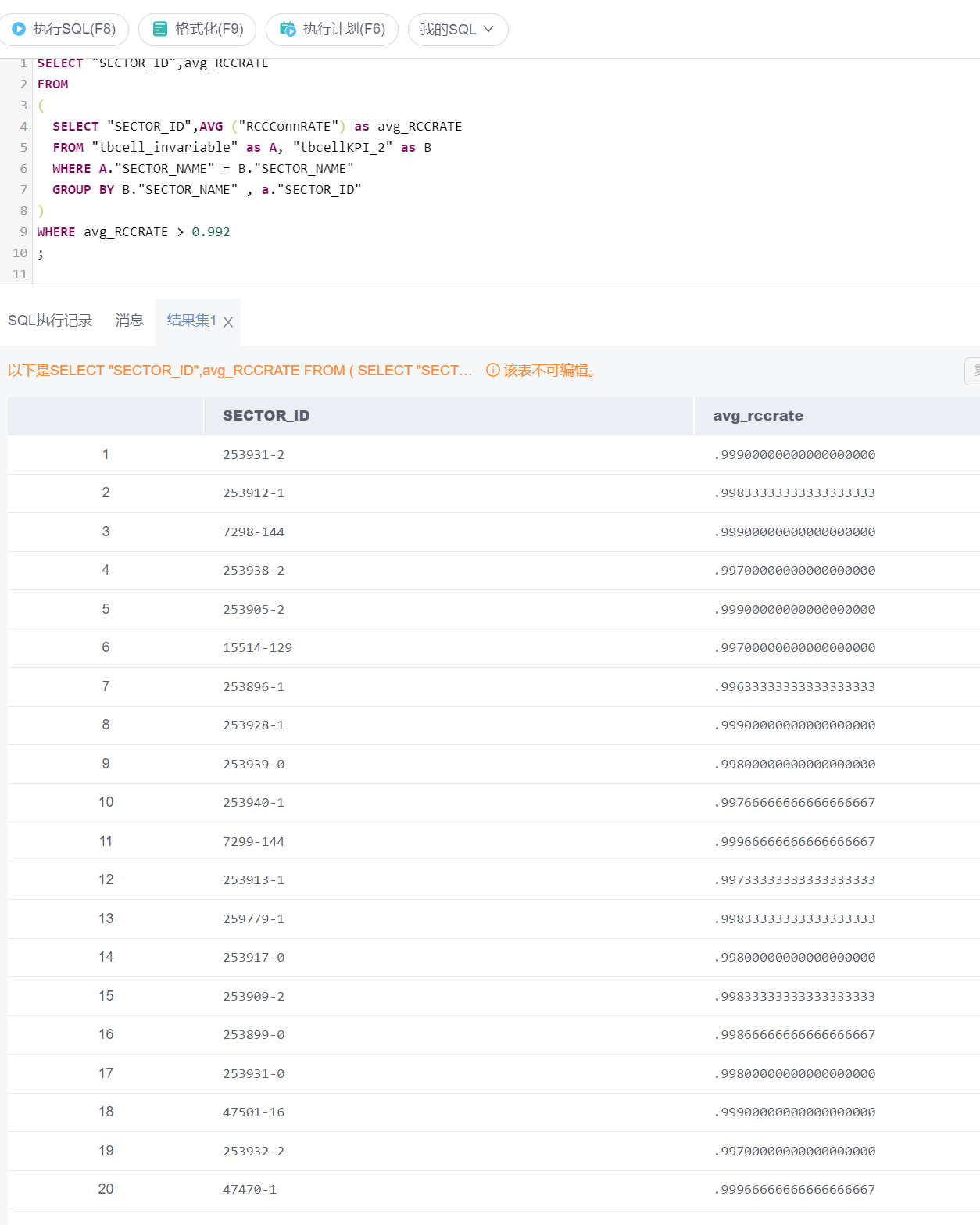
)

WHERE avg\_RCCRATE > 0.992

;

（3）查询结果

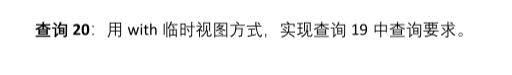




1. with临时视图查询

* 查询20

（1）查询内容



（2）查询语句

WITH avg\_RCC("SECTOR\_ID",avg\_RCCRATE) as

(

  SELECT "SECTOR\_ID",AVG ("RCCConnRATE") as avg\_RCCRATE

  FROM "tbcell\_invariable" as A, "tbcellKPI\_2" as B

  WHERE A."SECTOR\_NAME" = B."SECTOR\_NAME"

  GROUP BY B."SECTOR\_NAME" , a."SECTOR\_ID"

)

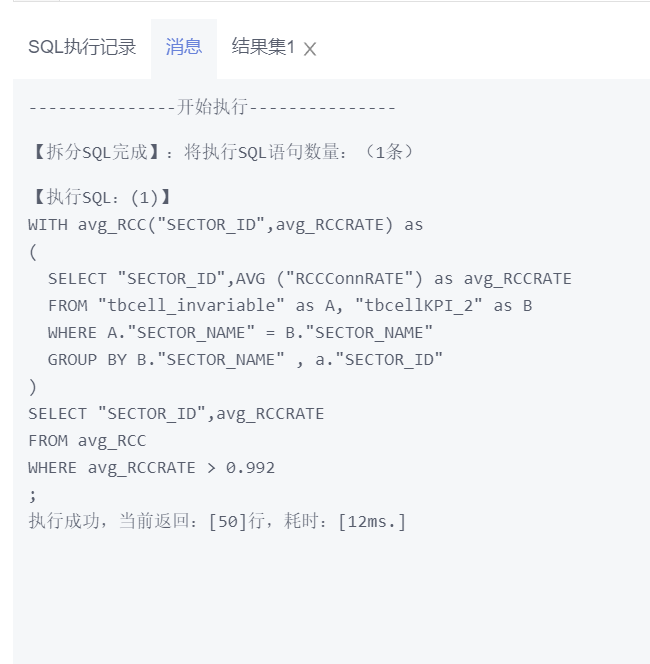
SELECT "SECTOR\_ID",avg\_RCC

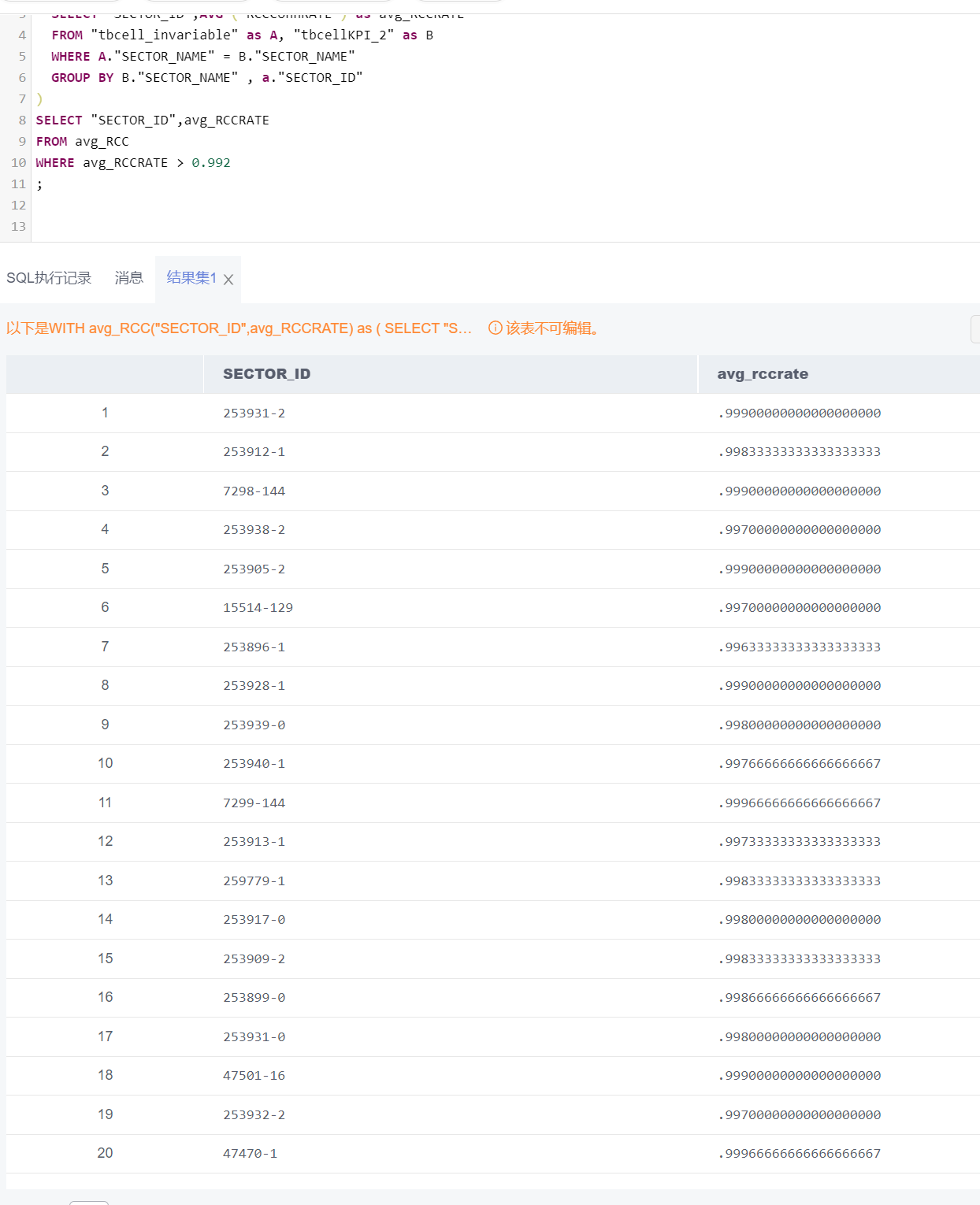
FROM avg\_RCCRATE

WHERE avg\_RCCRATE.value > 0.992

;

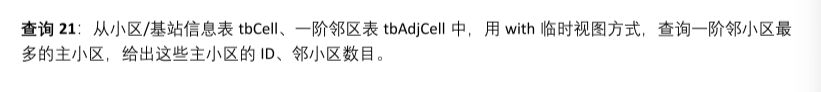
（3）查询结果





* 查询21

（1）查询内容



（2）查询语句

WITH cnt\_tb("S\_SECTOR\_ID",Ncnt) as

(

  SELECT "S\_SECTOR\_ID",COUNT ("N\_SECTOR\_ID")

  FROM "tbAdjCell"

  GROUP BY "S\_SECTOR\_ID"

  HAVING COUNT ("N\_SECTOR\_ID") >= all

  (

    SELECT COUNT ("N\_SECTOR\_ID")

    FROM "tbAdjCell"

    GROUP BY "S\_SECTOR\_ID"

  )

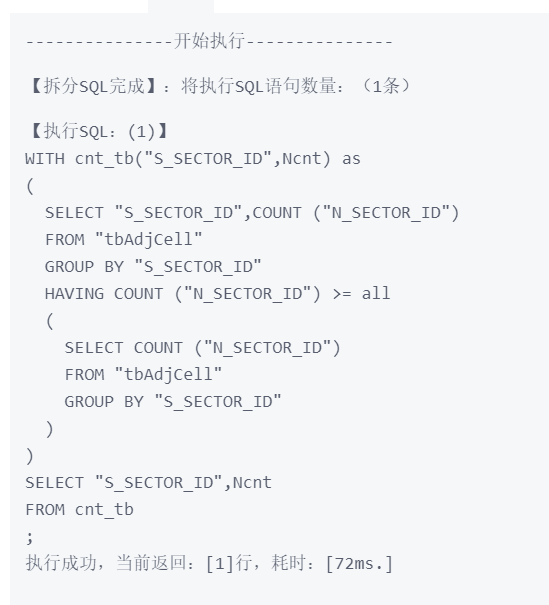
)

SELECT "S\_SECTOR\_ID",Ncnt

FROM cnt\_tb

;

（3）查询结果

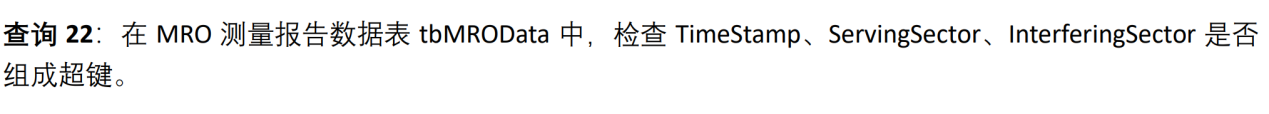




1. 键/函数依赖分析

* 查询22

（1）查询内容

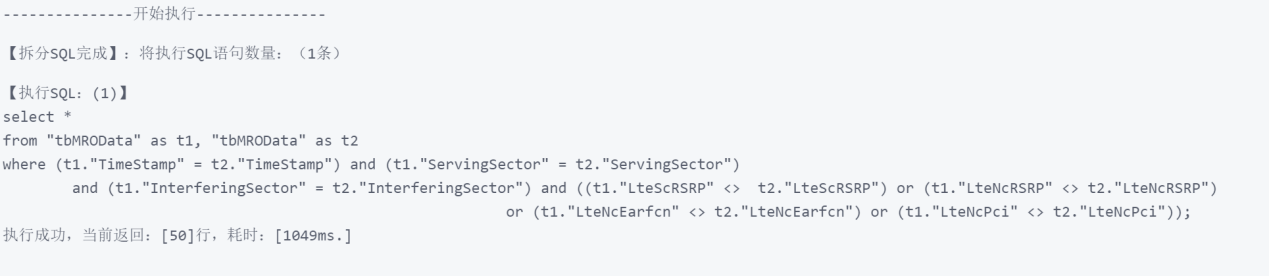


（2）查询语句

1. **select** \*
2. **from** "tbMROData" **as** t1, "tbMROData" **as** t2
3. **where** (t1."TimeStamp" = t2."TimeStamp") and (t1."ServingSector" = t2."ServingSector")
4. and (t1."InterferingSector" = t2."InterferingSector") and ((t1."LteScRSRP" <>  t2."LteScRSRP") or (t1."LteNcRSRP" <> t2."LteNcRSRP")
5. or (t1."LteNcEarfcn" <> t2."LteNcEarfcn") or (t1."LteNcPci" <> t2."LteNcPci"));

（3）查询结果

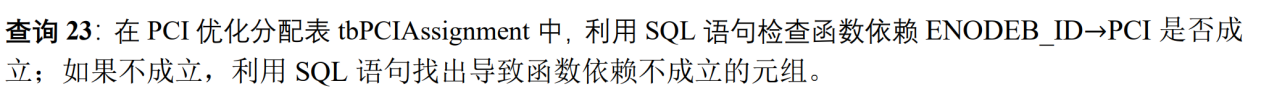
如图所示，结果元组非空，所以 TimeStamp、ServingSector、InterferingSector 不组成超键。





* 查询23

（1）查询内容



（2）查询语句

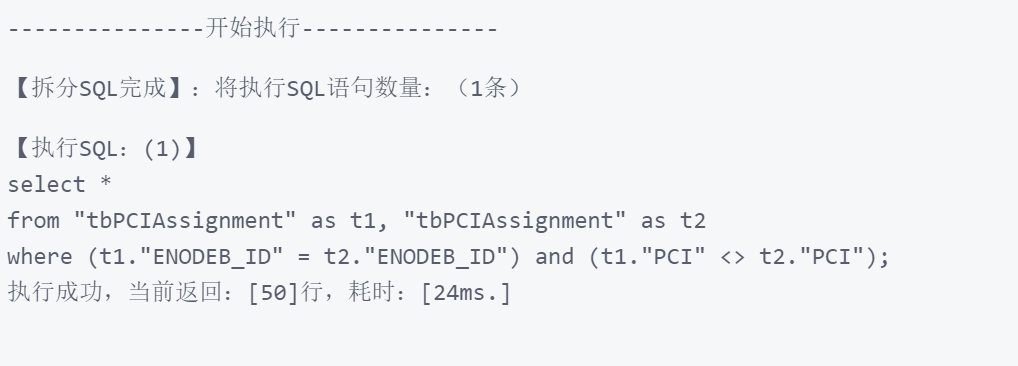
首先创建 tbPCIAssignment 表：

1. **create** **table** "tbPCIAssignment"
2. (
3. "ASSIGN\_ID" serial,
4. "EARFCN" **int**,
5. "SECTOR\_ID" **varchar**(50),
6. "SECTOR\_NAME" **varchar**(200),
7. "ENODEB\_ID" **int**,
8. "PCI" **int**,
9. "PSS" **int**,
10. "SSS" **int**,
11. "LONGITUDE" **float**,
12. "LATITUDE" **float**,
13. "STYLE" **varchar**(50),
14. "OPT\_DATETIME" **timestamp**,
15. **check** ("EARFCN" >= 37900),
16. **primary** **key** ("ASSIGN\_ID", "SECTOR\_ID")
17. );

寻找函数依赖不成立的元组：

1. **select** \*
2. **from** "tbPCIAssignment" **as** t1, "tbPCIAssignment" **as** t2
3. **where** (t1."ENODEB\_ID" = t2."ENODEB\_ID") and (t1."PCI" <> t2."PCI");

（3）查询结果

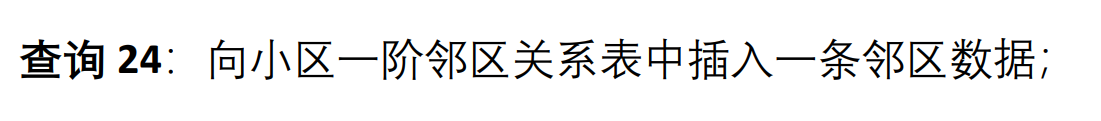




1. 表的插入、删除、更新

* 查询24

（1）查询内容

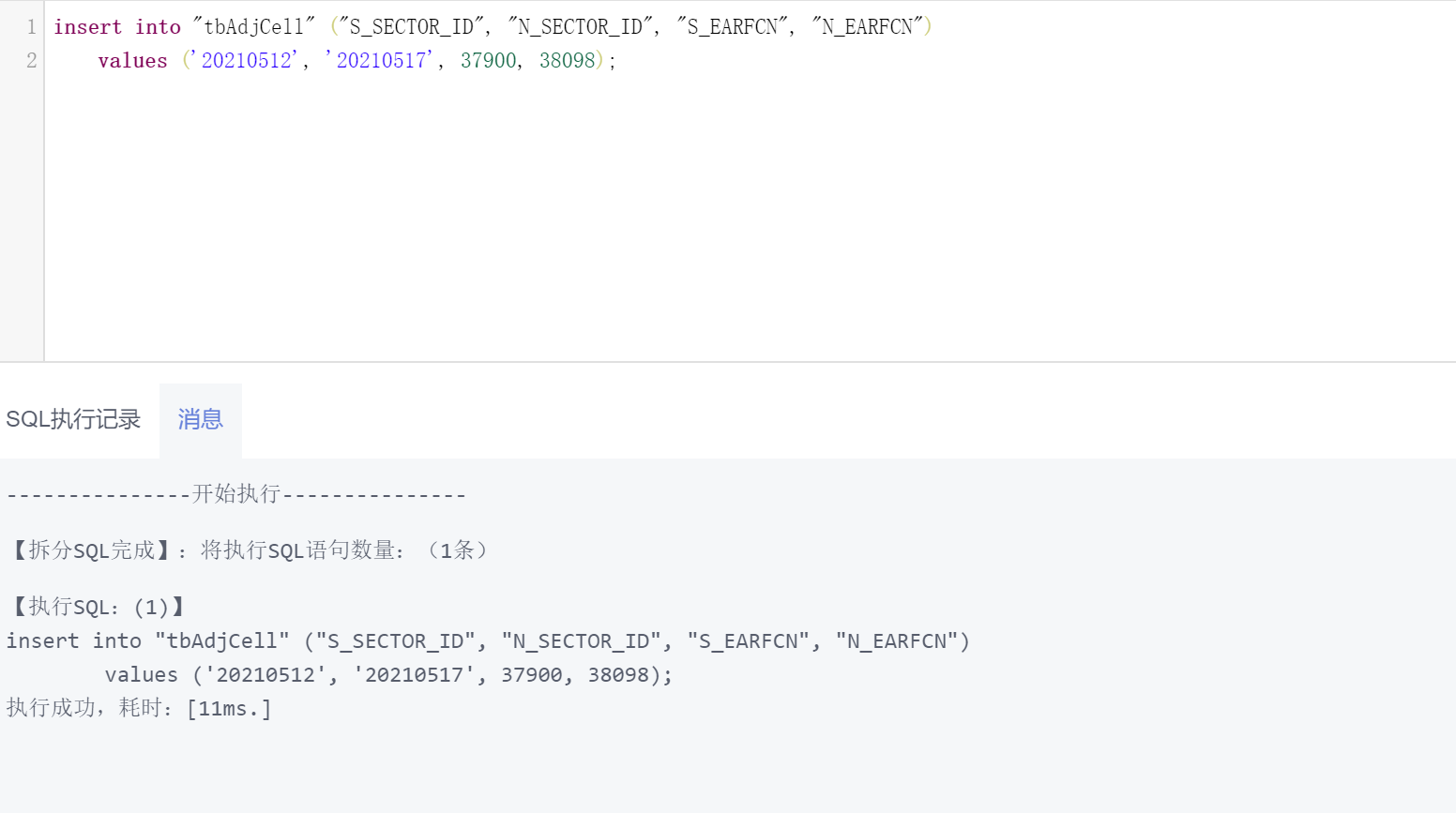


（2）查询语句

1. **insert** **into** "tbAdjCell" ("S\_SECTOR\_ID", "N\_SECTOR\_ID", "S\_EARFCN", "N\_EARFCN")
2. **values** ('20210512', '20210517', 37900, 38098);

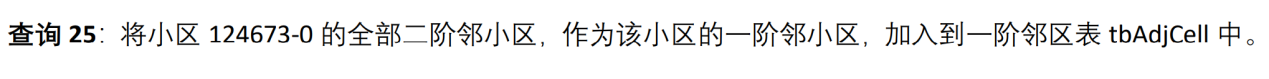
（3）查询结果

如图所示，插入成功。



* 查询25

（1）查询内容



（2）查询语句

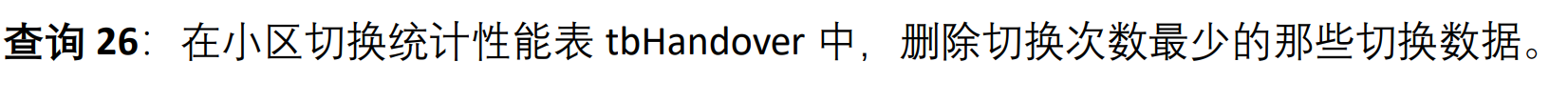
1. **insert** **into** "tbAdjCell"
2. **select** ts."S\_SECTOR\_ID", ts."N\_SECTOR\_ID", tc."EARFCN", tc."EARFCN"
3. **from** "tbSecAdjCell" **as** ts, "tbcell" **as** tc
4. **where** ts."S\_SECTOR\_ID" = tc."SECTOR\_ID";

（3）查询结果



* 查询26

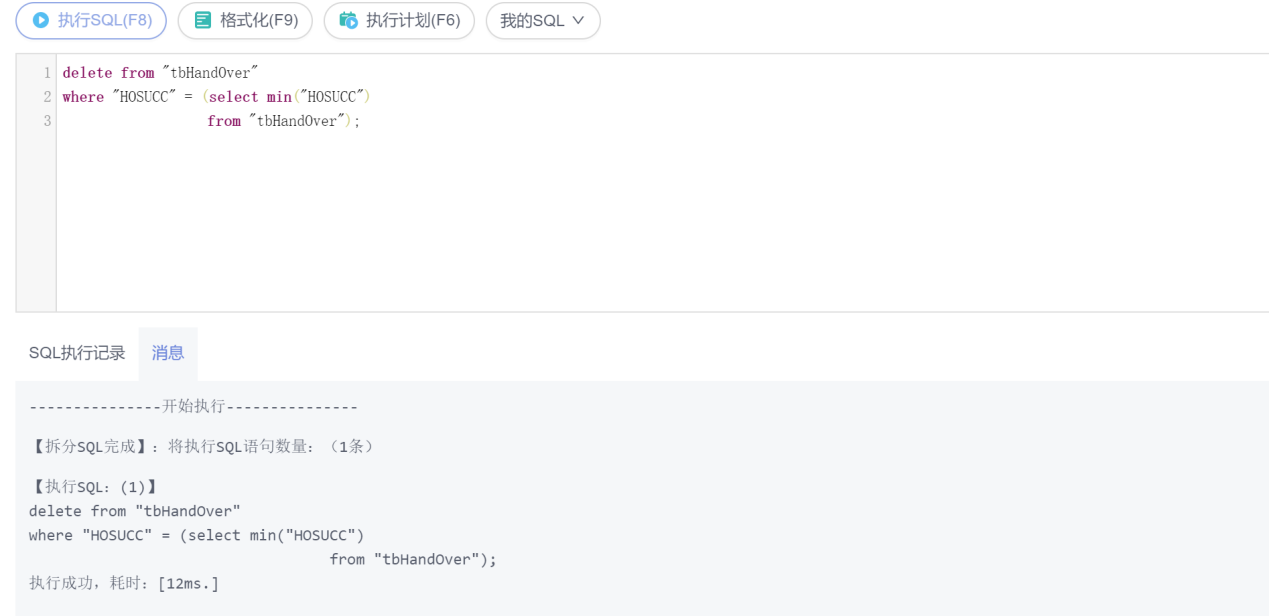
（1）查询内容



（2）查询语句

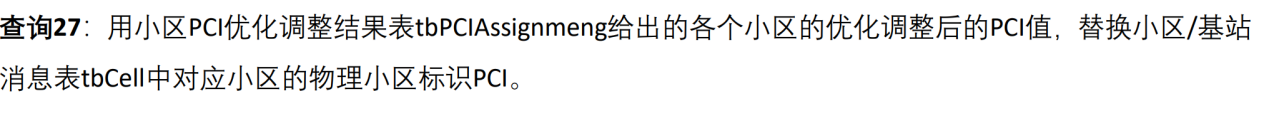
1. **delete** **from** "tbHandOver"
2. **where** "HOSUCC" = (**select** **min**("HOSUCC")
3. **from** "tbHandOver");

（3）查询结果



* 查询27

（1）查询内容



（2）查询语句

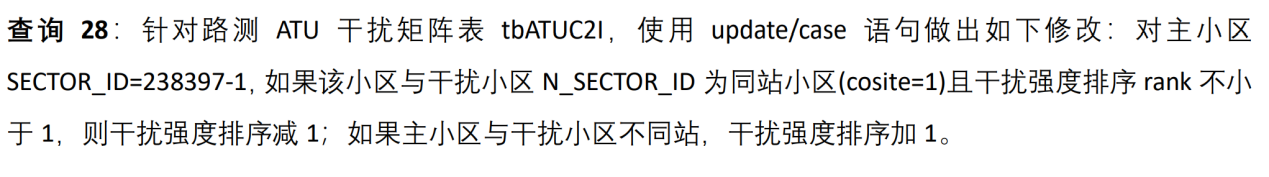
1. **update** "tbcell\_2" **as** tc
2. **set** "PCI" = case
3. **when** "SECTOR\_ID" in (**select** "SECTOR\_ID"
4. **from** "tbPCIAssignment")
5. **then** (**select** "PCI"
6. **from** "tbPCIAssignment"
7. **where** tc."SECTOR\_ID" = "tbPCIAssignment"."SECTOR\_ID")
8. **else** "PCI" \* 1.0
9. **end**

（3）查询结果



* 查询28

（1）查询内容



（2）查询语句

1. **update** "tbATUC2I"
2. **set** "RANK" = case
3. **when** "COSITE" = 1 and "RANK" >= 1 **then** "RANK" - 1
4. **when** "COSITE" = 0 **then** "RANK" + 1
5. **end**
6. **where** "SECTOR\_ID" = '238397-1';

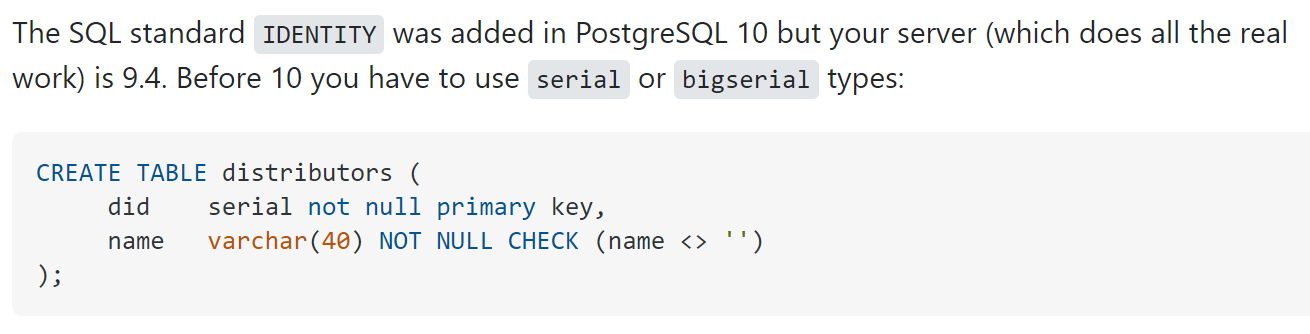
（3）查询结果



三、遇到问题及解决

1、版本问题

在建立 tbPCIAssignment 表的过程中，设置 ASSIGN\_ID 的自增首先采用 IDENTITY ，发现无法编译通过。查阅了 Stack Overflow 后发现，在 PostgreSQL 10 之前，自增只能使用 serial 或者 bigserial。



使用 select version() 后发现，当前数据库的 PostgreSQL 版本为 9.2.4，低于 10，改用 serial，问题得以解决。



2、delete测试

对于查询26，我首先考虑删除一个元组后，“=”后的 select 语句可能会发生改变，后来查阅了教科书，发现delete在执行删除操作之前，会进行所有元组的测试，这是至关重要的，问题得以解决。

1. **delete** from "tbHandOver"
2. where "HOSUCC" = (select min("HOSUCC")
3. from "tbHandOver");

3、not unique问题

对于查询18，运行not unique：



经过对postgre sql文档、stackoverflow的查询，我们发现not unique并没有在postgre sql中显式定义：

