第一章　完整性约束实验

【说明：罗列的实验内容比较多，不必都做。类似实验内容选做有代表性的，例如，

1） 主键验证、候选键验证只做一个；

2） 在一个实验中同时验证空值、默认值、主键、check 等约束；

3） 级联、非级联外键约束实验二选一

】

附：云数据库 GaussDB(for openGauss) SQL 参考

https://support.huaweicloud.com/devg-opengauss/opengauss_devg_0255.html

实验目的

了解 SQL 语言和 GaussDB 数据库提供的完整性（integrity）机制，通过实验掌握面向实际数据库建立实体完整性、参照完整性、断言、函数依赖等各种完整性约束的方法，验证各类完整性保障措施。

实验环境

采用 GaussDB(for openGauss)数据库管理系统作为实验平台。

实验内容

在前面完成的实验 1、2 中已建立了本实验所需的 14 张表。本实验将针对这 14 张表，采用 create table、alter table 等语句，添加主键、候选键、外键、check 约束、默认/缺省值约束，并观察当用户对数据库进行增、删、改操作时，DBMS 如何维护完整性约束。

1. 建立完整性约束
2. 主键/候选键/空值/check/默认值约束验证
3. 外键/参照完整性验证分析
4. 函数依赖
5. 触发器

实验步骤

依次完成以下各实验。

3.1 利用 Create table/Alter table 语句建立完整性约束

选择 LTE 网络数据库中的一张表，如小区/基站工参表 tbCell，分析识别该表上所有约束；采用 create table 语句，建立该表的副本 tbCellcopy，将数据导入 tbCellcopy，后续实验可在上进行。

（1）步骤 1. 使用 create table 在该表相关属性上，添加主键、唯一键、非空、默认/缺省值、check 等约束。示例：

```
create table tbCell (SECTORID nvarchar(50),
                     SECTORNAME nvarchar(255),
                     ENODEBID    int      not null,
                     …        ,
                     Azimuth float default 0,
                     primary key(SECTOR_ID),
                     unique key(SECTORNAME),
                     check (EARFCN in (37900, 38098,38400,38950,39148)),
                     check (PCI = 3*SSS + PSS),
                     …
```

表 3-1 tbCell 表上的约束（部分）

属性名称	字段中文名称	数据类型	完整性约束
SECTOR_ID	小区 ID	nvarchar(50)	主键
SECTORNAME	小区名称	nvarchar(255)	唯一键/候选键 unique key
ENODEBID	基站 ID	int	not null
EARFCN	小区频点编号	int	取值范围： (37900, 38098,38400,38950,39148)
PCI	物理小区标识 PHYCELLID	**int**	约束 1：PCI between 0 and 503 约束 2：PCI = 3*SSS + PSS
PSS	主同步信号标识	int	取值{0,1,2}；
SSS	辅同步信号标识	int	取值{0,1,2,…,167}；
LONGITUDE	小区所在基站经度	float	缺省值 112.77068
LATITUDE	小区所在基站纬度	float	缺省值 33.810396
AZIMUTH	小区天线方位角	float	缺省值 0
HEIGHT	小区天线高度	float	缺省值 20

（2）步骤 2. 使用 alter table 语句，在该表上添加约束。

- 新建主键，如果表中没有建立主键，利用下面的语句添加主键

  alter table 表名

  add constraint PK_字段名

  primary key (字段名)

  说明：PK 为主键的缩写，字段名为要在其上创建主键的字段名,'PK_字段名'就为约束名.

  示例: alter table tbCell

  add constraint PK_SECTORID

  primary key(SECTORID)

- 候选键，如果表中没有建立候选键，利用下面的语句添加候选键

  alter table 表名

  add constraint UQ_字段名

  unique (字段名)

  说明：UQ 为候选键的缩写

- 外键约束，如果表中没有建立外键，利用下面的语句添加外键

  alter table 表名

  add constraint FK_字段名

  foreign key (字段名) references 关联的表名(关联的字段名)

  说明：FK 为外键的缩写

- check 约束，利用下面语句在表中添加约束

  altertable 表名

  add constraint CK_字段名 check(表达式)

  示例：alter table tbCell

  add CK_PCI check(PCI=3*SSS + PSS)

- 默认/缺省值约束，利用下面语句在表中添加属性缺省值约束

  alter table 表名

  add constraint DF_字段名

  default 默认值 for 列名

  示例：alter table tbCell

  constraint DF_ LONGITUDE
  default 112.77068 for LONGITUDE

3.2 主键/候选键/空值/check/默认值约束验证

3.2.1 主键约束

选取定义了主键的关系表，如 tbCell、tbAdjCell、tbOptCell 等，

（1）使用分组聚集运算语句，判断是否满足主键约束

E.g. select SECTOR_ID, count(*)
from tbCell
group by SECTOR_ID
having count(*)>1
select * from tbCell where SECTOR_ID is null

（2）向该表插入在主属性上取值为空的元组，观察 DBMS 反应；

（3）选取表中某些或某个元组，修改这些元组在主属性上的取值，或向表中插入新元组，使这些元组与表中已有其它元组的主属性取值相同，或者将选定的元组在主属性上的取值修改为 null，观察 DBMS 反应；

3.2.2 候选键约束

（1）选取定义了候选键的关系表，如 tbCell、tbAdjCell、tbOptCell 等，使用分组聚集运算语句，判断是否满足候选键约束

E.g. select SECTOR_NAME, count(*)
from tbCell
group by SECTOR_NAME
having count(*)>1

（2）向该表插入在候选键属性上取值为空的元组，观察 DBMS 的反应；

（3）选取表中某些或某个元组，修改这些元组在候选键属性上的取值，或插入新元组，使这些元组与表中已有其它元组的候选键属性取值相同，或者将选定的元组在候选键属性上的取值修改为 null，观察系统反应；

示例：将 SSECTOR_ID，NSECTOR_ID 设置为 tbATUHandOver 的候选键，并修改某一个元组的该属性值为空；将 tbATUHandOver 表中 SSECTOR_ID 为 '15113-129'的记录中的 SSECTOR_ID 修改为 NULL。**比较在主键、候选键属性上插入 null 值或重复值时，DBMS 的不同反应和处理方式。**

3.2.3 空值

选取定义了 not null 属性约束的关系表，如 tbCell 及其属性 ENODEBID，观察

（1）向表中插入新元组，或者

（2）修改表中已有元组时，如果导致该属性上取值为空，DBMS 的反应和处理方式。

3.2.4 Check 约束

选取定义了 check 约束的关系表，如 tbCell，观察

（1）向表中插入新元组，或者

（2）修改表中已有元组时，如果违反表中已经定义的 check 约束，DBMS 的不同反应和处理方式。

示例：针对 tbCell 表中 PCI 属性上的 check 约束关系，修改表中某些行的 PCI 取值，或插入新元组，观察当两个 PCI 约束被违反时，DBMS 的反应和处理方式。

3.2.5 默认值

选取在属性上定义了默认值的关系表，如 tbCell 上的属性 LONGITUDE，以及默认值 112.77068，观察

（1）向表中插入在该属性上取值为 null 的新元组，或者

（2）将表中已有元组在在该属性上取值修改为 null 时，观察在插入或修改后的元组中，该属性上的取值是否为默认值。

3.3 外键/参照完整性约束验证

3.3.1 参照完整性约束验证

参照完整性约束要求：参照关系表 r_1 在**外键**上的全部属性取值，必须出现在被参照关系 r_2 的**主键**属性取值中。但在实际应用中，这种外键约束未必能够保证。

例如，小区/基站信息表 tbCell 表以 SECTOR_ID 为主键，记录了网络覆盖范围内一部分（并非全部）小区/基站的信息。邻区关系表 tbAdjCell 和 tbSecAdjCell 中作为外键的主小区和邻小区、切换关系表 tbHandover 中的作为外键的源小区和目标小区、小区 C2I 干扰表 tbC2I 中作为外键的主小区和干扰小区，均位于网络覆盖范围内，本应出现在 tbCell 表中。

但由于以下 2 个原因，这些作为外键属性的某些邻小区、切换源/目标小区、干扰小区并没有出现在 tbCell 表：

（1）tbCell 表中只记录了本地网络覆盖范围内的一部分小区/基站信息，并非全部；

（2）在本地网络覆盖边缘处，本地网络与外地网络重叠覆盖，tbAdjCell、tbSecAdjCell、tbHandover、tbC2I 记录的小区可能来自外地网络，在 tbCell 表中无记录。

表 2 可能存在外键关联的表

参照关系 r_1		被参照关系 r_2		
表	主键	表	外键属性 1	外键属性 2
tbCell	SECTOR_ID	tbAdjCell	S_SECTOR_ID	N_SECTOR_ID
tbCell	SECTOR_ID	tbSecAdjCell	S_SECTOR_ID	N_SECTOR_ID

tbCell	SECTOR_ID	tbHandover	SCELL	NCELL
tbCell	SECTOR_ID	tbC2I	SCELL	NCELL

实验过程：

步骤 1：判断参照完整性约束是否满足

从表 2 中选定一组表 r_1 和 r_2，编写 SQL 语句，判断两表间是否满足参照完整性约束。例如： r_1=tbCell，r_2=tbAdjCell，判断 tbAdjCell 在属性 N_SECTOR_ID 上的取值是否都出现在 tbCell 表的 SECTOR_ID 列中。

```
E.g.    select    N_SECTOR_ID
        from    tbAdjCell
        where    N_SECTOR_ID not in { select    SECTOR_ID
                                      from    tbCell
                                    }
```

步骤 2：改造参照关系表，满足完整性要求

如果上述 SQL 语句查询结果不为空，说明两张表间不满足参照完整性约束。使用 delete 语句，去除参照关系表中相关元组，如上述 SQL 查询语句中查出的 N_SECTOR_ID 所在元组，使得两表间参照完整性约束关系成立。

在改造后的参照表 r_1 和被参照 r_2 上，建立非级联外键关联，例如

```
E.g.    Alter table tbAdjCell
        Add constraint FK_ N_SECTOR_ID
        foreign key(N_SECTOR_ID) references tbCell (SECTOR_ID)
```

3.3.2 非级联外键关联下数据访问

选取相互间定义了非级联外键关联的一组表 r_1 和 r_2，分别在参照关系 r_1、被参照关系 r_2 上，对表的主属性/外键属性作插入 insert、删除 delete、更新 update 操作，观察当其中 1 个表（如参照关系表 r_1、被参照关系表 r_2）在外键属性或主属性上的取值发生变化时，DBMS 对这些操作的反应，以及另外一个表（如被参照关系表、参照关系表）在主属性或外键属性上的取值的变化，并记录实验结果。

上述插入、删除、更新操作操作分为违反约束和不违反约束两种情况。

3.3.3 级联外键关联下数据访问

步骤 1：使用 `Alter table` 中的 `Drop constraint` 参数，删除前面定义的参照关系 r_1 和被参照关系 r_2 间的非级联关联，重新定义级联关联：

```
Alter table tbAdjCell
```

```
Drop constraint FK_ N_SECTOR_ID

Alter table tbAdjCell
Add constraint FK_ N_SECTOR_ID
Foreign key(N_SECTOR_ID) references tbCell (SECTOR_ID)
on delete cascade
on update cascade
```

步骤 2：分别在参照关系 r_1、被参照关系 r_2 上，对表的主属性/外键属性作插入 insert、删除 delete、更新 update 操作，观察当其中 1 个表（如参照关系表 r_1、被参照关系表 r_2）在外键属性或主属性上的取值发生变化时，DBMS 对这些操作的反应，以及另外一个表（如被参照关系表、参照关系表）在主属性或外键属性上的取值的变化，并记录实验结果。

上述插入、删除、更新操作操作分为违反约束和不违反约束两种情况。

3.4 函数依赖分析验证【参考第 8 章作业题 8.9】

函数依赖反映了关系表中属性间的依赖关系。主键、候选键、外键约束都属于函数依赖，对于这三类函数依赖的验证参见实验 3.3.1、3.3.2、3.3.3。下面考虑验证非主属性间的函数依赖关系。

实验 1：

在小区/基站信息表 tbCell 中，同属于一个基站的各个小区的经纬度是一样的，因此基站 ID 与小区/基站经纬度间存在函数依赖：

ENODEBID → LONGITUDE, LATITUDE

要求：

（1）用 SQL 语句判断 ENODEBID 与 LONGITUDE, LATITUDE 间是否存在函数依赖关系。

（2）如果 ENODEBID 与 LONGITUDE, LATITUDE 间函数依赖不存在，用 SQL 语句找出导致该函数依赖不存在的元组。

```
E.g.  select   ENODEBID, T1.LONGITUDE,
                  T2.LONGITUDE, T1.LATITUDE, T2.LATITUDE
       from   tbCell as T1, tbCell as T2
       where T1. ENODEBID= T2. ENODEBID
                  and (T1.LONGITUDE<> T2.LONGITUDE OR T1.LATITUDE<> T2.LATITUDE)
```

3.5 触发器约束【选做一个】

实验 1：

开发一个数据插入查重触发器，实现：

向一张表中插入一行新数据时，如果新数据的主键与表中已有其它元组的主键不相同，则直接插入；如果新数据的主键与表中已有元组的主键相同，则根据新插入元组的属性值修改已有元组的属性值，或者：先删除主键相同的已有元组，再插入新元组。

实验 2：

开发一个 PCI 修改触发器，实现：

当向小区/基站信息表 tbCell 中插入一行，或者修改现有小区的 PCI 值时，判断新插入的、或修改后的小区 PCI 是否合法，即 PCI 是否在合法范围 0~503 内。如果不合法，回滚；对于合法的 PCI 值，计算 SSS=PCI/3【除 3 向下取整】，PSS= PCI mod 3，设置小区的 SSS、PSS 值。

实验 3：

开发一个实时更新小区天级话务统计信息的触发器，实现根据小区小时级话务数据 traffic 的变化动态更新小区的天级话务统计数据。

步骤 1. 根据已建立的数据表“tbCell_traffic-57 个小区一年小时级数据 traffic”，建立统计小区每天 24 小时话务数据 traffic 总和的小区天级话务数据表 tbCell_traffic_Day;

步骤 2. 建立“小区天级话务统计信息更新”触发器，实现：

当向表“tbCell_traffic-57 个小区一年小时级数据”中插入一条某小区的小时级话务数据，或更新数据库中已有的该小区小时级话务数据时，该触发器自动更新表 tbCell_traffic_Day 中该小区的天级话务统计数据。

步骤 3. 向表“tbCell_traffic-57 个小区一年小时级数据”中插入数据、更新已有数据，观察表 tbCell_traffic_Day 中对应小区的天级话务数据的变化情况。

实验要求

1. 用 Transact_SQL 语句完成以上操作。
2. 要求学生独立完成以上内容。
3. 实验完成后完成要求的实验报告内容。

实验总结