openGauss 数据库开发查询实验

姓名：辛浩然 学号：2112514

实验步骤：

- 创建和管理用户、表空间和数据库
- 创建和管理表
- 创建和管理其他数据库对象
- 学校数据模型创建及表操作

实验报告

实验步骤截图：

截图 1：指导手册第 8 页，查询表空间当前使用情况截图

```
postgres=# SELECT PG_TABLESPACE_SIZE('fastspace');
 pg_tablespace_size
--------------------
               4096
(1 row)
```

截图 2：指导手册第 10 页，创建表截图

```
postgres=# CREATE TABLE customer_t1
postgres-# (
postgres(#     c_customer_sk            integer,
postgres(#     c_customer_id            char(5),
postgres(#     c_first_name             char(6),
postgres(#     c_last_name              char(8)
postgres(# );
CREATE TABLE
```

截图 3：指导手册第 16 页，向分区表中插入数据后查看分区表中所有数据并截图（该命令需自行撰写）

```
select * from tpcds.web_returns_p2;
```

```
postgres=# select * from tpcds.web_returns_p2;
 ca_address_sk |  ca_address_id   | ca_street_number | ca_street_name | ca_street_type  | ca_suite_number | ca_city | ca_county | ca_state |   ca_zip   | ca_country | ca_gmt_offset |
  ca_location_type
---------------+------------------+------------------+----------------+-----------------+-----------------+---------+-----------+----------+------------+------------+---------------+--
--------------------
             1 | a                | 1                | a              | a               | a               | a       | a         | a        | a          | a          |          1.00 |
a
             2 | b                | 2                | b              | b               | b               | b       | b         | b        | b          | b          |          1.10 |
b
          5050 | c                | 300              | c              | c               | c               | c       | c         | c        | c          | c          |          1.20 |
c
         14888 | d                | 400              | d              | d               | d               | d       | d         | d        | d          | d          |          1.50 |
d
(4 rows)
```

截图 4：指导手册第 19 页，创建分区索引截图。

```
postgres=# CREATE INDEX tpcds_web_returns_p2_index2 ON tpcds.web_returns_p2 (ca_addre
ss_sk) LOCAL
postgres-# (
postgres(#     PARTITION web_returns_p2_P1_index,
postgres(#     PARTITION web_returns_p2_P2_index TABLESPACE example3,
postgres(#     PARTITION web_returns_p2_P3_index TABLESPACE example4,
postgres(#     PARTITION web_returns_p2_P4_index,
postgres(#     PARTITION web_returns_p2_P5_index,
postgres(#     PARTITION web_returns_p2_P6_index,
postgres(#     PARTITION web_returns_p2_P7_index,
postgres(#     PARTITION web_returns_p2_P8_index
postgres(# ) TABLESPACE example2;
CREATE INDEX
```

截图 5：指导手册第 23 页，更新物化视图。

```
postgres=# SELECT * FROM MV_MyView;
 ca_address_sk |  ca_address_id   | ca_street_number | ca_street_name | ca_street_type  | ca_s
uite_number | ca_city | ca_county | ca_state |   ca_zip   | ca_country | ca_gmt_offset |   ca_
location_type
---------------+------------------+------------------+----------------+-----------------+-----
------------+---------+-----------+----------+------------+------------+---------------+------
---------------
          5050 | c                | 300              | c              | c               | c
            | c       | c         | c        | c          |            |          1.20 | c

          7050 | c                | 300              | c              | c               | c
            | c       | c         | c        | c          |            |          1.20 | c

          8888 | d                | 400              | d              | d               | d
            | d       | d         | d        | d          |            |          1.50 | d

         14888 | d                | 400              | d              | d               | d
            | d       | d         | d        | d          |            |          1.50 | d

(4 rows)
```

截图 6：指导手册第 26 页，管理存储过程

```
postgres=# \sf insert_data
CREATE OR REPLACE PROCEDURE public.insert_data()
AS  DECLARE
a int;
b int;
begin
a=1;
b=2;
insert into t_test values(a,b);
insert into t_test values(b,a);
end;
/
```

截图 7：指导手册第 39 页，删除数据后表中内容截图

```
postgres=# DELETE FROM school_department WHERE depart_teacher=8 OR depart_teacher=15;
DELETE 2
postgres=# SELECT * FROM school_department;
 depart_id |         depart_name         | depart_teacher
-----------+-----------------------------+----------------
         1 | 计算机学院                  |              2
         2 | 自动化学院                  |              4
         3 | 航空宇航学院                |              6
         5 | 理学院                      |             11
         6 | 人工智能学院                |             13
         8 | 管理学院                    |             17
         9 | 农学院                      |             22
        10 | 医学院                      |             28
(8 rows)
```

实验思考题：

1. 在 openGauss 中，创建具有“创建数据库”权限的用户 Alice，并设置其初始密码为“openGauss@0331”,应使用的语句是：

   `CREATE USER Alice CREATEDB PASSWORD 'openGauss@0331';`

2. 命令 “DROP USER kim CASCADE”的效果是？（可以预习参考第八周主讲课内容，权限和授权）

   删除用户 kim 及 kim 名下所有的对象。如果用户的 schema 中有表，则在删除表的时候自动删除与该表相关的主键和外键，自动删除与该表相关的主键和外键。其他用户建立的基于被删除用户的物化视图不会被删除，只是不能再刷新。被删除用户建立的其他用户不会被删除。

3. 向表中插入数据时，是否允许只对部分属性插入数值？在何种情况下允许，应如何书写语句？何种情况下不允许？

   允许只对部分属性插入数值。需要未插入数值的属性非键属性，该字段将被填充为字段的缺省值。

   如果用户不知道所有字段的数值，可以忽略其中的一些。没有数值的字段将被填充为字段的缺省值。

   `# insert into` 表名（需要插入数值的属性）`values`（对应插入的数值）

   `INSERT INTO customer_t1 (c_customer_sk, c_first_name) VALUES (3769, 'Grace');`

   如果用户已经知道表中字段的顺序，也可无需列出表中的字段。没有数值的字段将被填充为字段的缺省值。

insert into 表名（按顺序输入插入的属性数值）

```
INSERT INTO customer_t1 VALUES (3769, 'hello');
```

用户也可以对独立的字段或者整个行明确缺省值：

```
INSERT INTO customer_t1 DEFAULT VALUES;
```

```
INSERT INTO customer_t1 (c_customer_sk, c_customer_id, c_first_name) VALUES (3769, 'hello', DEFAULT);
```

主键值不允许为空，主键属性必须插入数值。

4. 是否可以向表中一次性插入多条数据？何种插入效率较高？

可以，一次性插入多条数据的效率较高。如果多次执行插入一行数据命令，多次调用 sql 语句，意味着多次与数据库建立连接。但是这样一来，就会增加服务器的负荷，因为，执行每一次 SQL 服务器都要同样对 SQL 进行分析、优化等操作。

5. openGauss 中将表中所有元组删除的两种命令是？

```
DELETE FROM 表名;
```

或者 TRUNCATE TABLE 表名;

6. 如果经常需要查询某字段值小于某一指定值的信息，可以如何操作？（提示，从索引角度思考）

为该查询创建表达式索引：

```
CREATE INDEX 索引 ON （表达式）
```

如假如经常需要查询 ca_street_number 小于 1000 的信息，执行如下命令进行查询：

```
SELECT * FROM tpcds.customer_address_bak WHERE trunc(ca_street_number) < 1000;
```

可以为上面的查询创建表达式索引：

```
CREATE INDEX para_index ON tpcds.customer_address_bak (trunc(ca_street_number));
```

7. 在什么场景下可以使用物化视图？物化视图和普通视图的区别是？

物化视图使用场景：报表统计、大表统计等，定期固化数据快照，避免对多表重复跑相同的查询。

物化视图使用注意事项：不可以在临时表或全局临时表上创建。当基表数据发生变化时，需要使用刷新命令保持物化视图与基表同步。

物化视图与普通视图区别在于：普通视图是不存储任何数据的，它只有定义，其真实数据在基表中，即每次查询视图都是需要执行查询语句。在而物化视图是将数据转换为一个表，实际存储着数据，这样查询数据，就不用关联一大堆表，如果表很大的话，会在临时表空间内做大量的操作。

8. 学校模型 ER 图绘制