openGauss 数据库维护管理实验

姓名：辛浩然 学号：2112514

实验步骤：

- openGauss 数据库安装
- 数据库性能检查实验
- 最大连接数设置实验
- 例行表、索引维护实验

实验报告

实验步骤截图：

截图 1：指导手册 25 页顺序扫描执行计划截图

```
postgres=# explain select * from student where std_id=30;
                        QUERY PLAN
-----------------------------------------------------------
 Seq Scan on student  (cost=0.00..1.62 rows=1 width=62)
   Filter: (std_id = 30)
(2 rows)
```

截图 2：指导手册 26 页索引扫描执行计划截图

```
postgres=# explain select /*+indexscan(student student_pkey)*/ * from student where std_id=30;
                                QUERY PLAN
---------------------------------------------------------------------------
 [Bypass]
 Index Scan using student_pkey on student  (cost=0.00..8.27 rows=1 width=62)
   Index Cond: (std_id = 30)
(3 rows)
```

截图 3：将最大连接数设置为 8000 并验证设置是否成功截图（注意，指导手册中将最大连接数设置为 6000，怎样重新设置为 8000 呢？）

用 alter system set 语句将参数修改为 8000，重启数据库。

```
postgres=# SHOW max_connections
postgres-# ;
 max_connections
-----------------
 6000
(1 row)

postgres=# alter system set max_connections=8000;
NOTICE:  please restart the database for the POSTMASTER level parameter to take effect.
ALTER SYSTEM SET
postgres=# \q
[omm@opengauss01 ~]$ gs_om -t stop;
Stopping cluster.
=========================================
Successfully stopped cluster.
=========================================
End stop cluster.
[omm@opengauss01 ~]$  gs_om -t start;
Starting cluster.
=========================================
[SUCCESS] opengauss01
2023-03-17 13:34:40.871 6413fbf0.1 [unknown] 281469512908816 [unknown] 0 dn_6001 01000
2023-03-17 13:34:40.874 6413fbf0.1 [unknown] 281469512908816 [unknown] 0 dn_6001 01000
cstore_buffers (16 Mbytes) or shared memory (2363 Mbytes) is larger.
=========================================
Successfully started.
[omm@opengauss01 ~]$  gsql -d postgres -p 26000 -r
gsql ((openGauss 2.0.0 build 78689da9) compiled at 2021-03-31 21:03:52 commit 0 last mr
Non-SSL connection (SSL connection is recommended when requiring high-security)
Type "help" for help.

postgres=#  SHOW max_connections;
 max_connections
-----------------
 8000
(1 row)
```

截图四：使用 ANALYZE VERBOSE 语句更新统计信息，并输出表的相关信息。(该步骤截图)

```
postgres=# analyze verbose student;
INFO:  analyzing "public.student"(dn_6001 pid=40500)
INFO:  ANALYZE INFO : "student": scanned 1 of 1 pages, containing 30 live rows and 20 dead rows; 30 r
mated total rows(dn_6001 pid=40500)
ANALYZE
```

实验思考题：

1. 全表扫描和索引扫描的区别是什么？具体是如何实现的？比较两种扫描方式的 cost（提供查询结果截图）,为什么全表扫描比索引扫描 cost 更小？在什么情况下通过主键进行查找会比全表扫描更节省时间？

   全表扫描是指扫描表中的每一行记录。实现方式：将数据从磁盘上一个个读到内存中做过滤，最后返回结果。

   索引扫描并不是顺序地获取所有记录，而是使用索引来查询表中数据。索引是一种数据结构，通过索引扫描可以只查询表中部分数据。实现方式：对磁盘上的数据建一个索引，并在内存中维护索引，索引将所有数据排序，并记录对应的磁盘位置。在查询时，首先在索引上过滤出所有结果集在磁盘上的

位置，再到磁盘上去精确读取结果集。

```
postgres=# explain select * from student where std_id=30;
                          QUERY PLAN
---------------------------------------------------------------
 Seq Scan on student  (cost=0.00..1.62 rows=1 width=62)
   Filter: (std_id = 30)
(2 rows)
```

```
postgres=# explain select /*+indexscan(student student_pkey)*/ * from student where std_id=30;
                                QUERY PLAN
---------------------------------------------------------------------------
 [Bypass]
 Index Scan using student_pkey on student  (cost=0.00..8.27 rows=1 width=62)
   Index Cond: (std_id = 30)
(3 rows)
```

在该实验中，数据库包含的数据量少，相较于索引扫描而言，全表扫描的 cost 更小。

数据库表中的数据非常大时，且查询条件可以通过主键快速定位到所需数据时，通过主键查找更节省时间。

2. **请列举一种需要重建索引的情况和原因，并说明 openGauss 中重建索引的方式有哪些。**

当对索引所在的基础数据表进行增删改时，若存储的数据进行了不适当的跨页（SQL Server 中存储的最小单位是页，页是不可再分的），就会导致索引碎片的产生。随着索引碎片的不断增多，查询响应时间就会变慢，性能也因此而下降。因而需要重建索引。

在 openGauss 中，重建索引的方式有以下几种：

方式 1：使用 REINDEX 语句重建索引，具体如下。

postgres=# reindex table student;

REINDEX

方式 2：先删除索引（DROP INDEX），再创建索引（CREATE INDEX），具体如下。

postgres=# drop index inx_stu01;

DROP INDEX

postgres=# create index inx_stu01 on student(std_name);

CREATE INDEX