Postgresql运维

统计

统计

PostgreSQL统计表行数

根据网上的资料修改脚本

```
#!/bin/bash
# Directory of the current script
CMD_DIRS=$(dirname $(readlink -f "$0"))
# Home
CMD_HOME=$(dirname ${CMD_DIRS})
# System home
SYS_HOME=$(cd ${CMD_HOME}/../../ && pwd)
if [[ -f "${CMD_HOME}/bin/setenv.sh" ]]; then
  . ${CMD_HOME}/bin/setenv.sh
fi
# Set defaults for configuration variables
PGSQL_HOME=${PGSQL_HOME:-"/usr/pgsql-${PGSQL_MAJOR_VERSION}"}
# Base directory
PGSQL_BASE=${CMD_HOME}
# Postgres user
PGSQL_USER=${PGSQL_USER:-"postgres"}
# Postgres role
PGSQL_ROLE=${PGSQL_ROLE:-"${PGSQL_USER}"}
# Postgres engine directory
PGSQL_ENGINE=${PGSQL_HOME}/bin
PGSQL_OPTS=${PGSQL_OPTS:-""}
# For SELinux we need to use 'runuser' not 'su'
if [[ -x "/sbin/runuser" ]]; then
    SU="/sbin/runuser -s /bin/bash"
else
    SU="/bin/su -s /bin/bash"
fi
# 使用操作系统账号免登录
if [[ $(whoami) != "${PGSQL_USER}" ]]; then
  SUDO="${SU} -l ${PGSQL_USER}"
  if [[ -x /bin/sudo && $(id -u) -ne 0 ]]; then
    SUDO="/bin/sudo ${SUDO}"
  fi
else
  SUDO="/bin/bash"
fi
# 统计数据库名称
PGSQL_DB_NAME=${1:-"snapshot"}
PGSQL_COUNT_OPTS="${PGSQL_OPTS} -d ${PGSQL_DB_NAME}"
if ! ${SUDO} -c "${PGSQL_ENGINE}/psql -qAtX ${PGSQL_COUNT_OPTS} -c \"SELECT VERSION()\" 2>&1 > /dev/null"; then
  echo "数据库查询失败"
  exit 1
fi
PGSQL_SQL_WHERE=""
# 获取服务器版本
PGSQL_CMD_OUT=$(${SUDO} -c "${PGSQL_ENGINE}/psql -qAtX ${PGSQL_COUNT_OPTS} -c \"SELECT VERSION()\"" | awk '{print $2}')
PGSQL_SERVER_VERSION=${PGSQL_CMD_OUT}
# postgresql-11分区表，排除主表防止行数重复
if [[ $(echo "${PGSQL_SERVER_VERSION} >= 11.0" | bc) -eq 1 ]]; then
  PGSQL_SQL_WHERE=" and not exists (SELECT * FROM (SELECT DISTINCT INHPARENT::REGCLASS AS TABLENAME FROM PG_INHERITS) AS T
WHERE TABLENAME = (SELECT OID FROM PG_CLASS WHERE RELNAME = pg_tables.tablename))"
fi
# 统计每个表的行数，以及总数
PGSQL_COUNT_SQL=$(cat <<EOF
SELECT CASE
          WHEN t.row_total = 1 THEN
           'select * from(' || REPLACE(sql_content, ' union all ', ') a order by 2 desc,3 desc')
          WHEN t.row_total = t.row_seq THEN
           REPLACE(sql_content, ' union all ', ') a order by 2 desc,3 desc')
          WHEN t.row_seq = 1 THEN
           'select * from(' || sql_content
```

```
          ELSE
           sql_content
       END sql_content
  FROM (SELECT COUNT(*) over() row_total,
               row_number() over() row_seq,
               'SELECT ''' || quote_ident(tablename) ||
               ''' 表名, count(*) 表行数,pg_total_relation_size(''' ||
               quote_ident(tablename) || '''::regclass) 表总大小 from ' || quote_ident(tablename) ||
               ' union all ' sql_content
          FROM pg_tables
         WHERE schemaname = 'public'${PGSQL_SQL_WHERE}) t
 ORDER BY t.row_seq
EOF
)
## 生成SQL
PGSQL_CMD_OUT=$(${SUDO} -c "${PGSQL_ENGINE}/psql -qAtX ${PGSQL_COUNT_OPTS} -c \"${PGSQL_COUNT_SQL}\"")
# 生成的SQL可能超长，需要使用"heredoc"方式
${SUDO} <<EOF
${PGSQL_ENGINE}/psql ${PGSQL_COUNT_OPTS} <<SQL
SELECT SUM(表行数) AS 表行数,pg_size_pretty(SUM(表总大小)) AS 表总大小 FROM (${PGSQL_CMD_OUT}) AS T
SQL
EOF
${SUDO} <<EOF
${PGSQL_ENGINE}/psql ${PGSQL_COUNT_OPTS} <<SQL
SELECT 表名, 表行数,pg_size_pretty(表总大小) AS 表总大小 FROM (${PGSQL_CMD_OUT}) AS T ORDER BY T.表总大小 desc, T.表行数 desc
SQL
EOF
```

参考资料

- 墨天轮 - 一键查询PostgreSQL模式中所有表行数和表大小

统计

Postgresql硬盘空间查询

一、查看所有表，索引占用空间

```
-- 表、索引占用空间
SELECT
    table_name,
    pg_size_pretty(table_size) AS table_size,
    pg_size_pretty(indexes_size) AS indexes_size,
    pg_size_pretty(total_size) AS total_size
FROM (
    SELECT
        table_name,
        pg_table_size(table_name) AS table_size,
        pg_indexes_size(table_name) AS indexes_size,
        pg_total_relation_size(table_name) AS total_size
    FROM (
        SELECT ('"' || table_schema || '"."' || table_name || '"') AS table_name
        FROM information_schema.tables
    ) AS all_tables
    ORDER BY total_size DESC
) AS pretty_sizes;
```

二、查看数据库占用空间

```
-- 查询指定数据库占用空间
select pg_size_pretty (pg_database_size('test_database'));

-- 查询所有数据库占用空间
select datname, pg_size_pretty (pg_database_size(datname)) AS size from pg_database;
```

三、查看当前数据库所有表占用空间

```
-- 查询当前数据库所有表占用空间
select
    table_full_name,
    pg_size_pretty(size)
from
(
        SELECT
            table_schema || '.' || table_name AS table_full_name,
            pg_total_relation_size(
                '"' || table_schema || '"."' || table_name || '"'
            ) AS size
        FROM
            information_schema.tables
        ORDER BY
            size DESC
    ) as T
```

等保测评

等保测评

Postgresql设置密码复杂度策略

安装和配置 passwordcheck 扩展

以下是如何使用 `passwordcheck` 扩展来设置 PostgreSQL 用户密码复杂度策略的步骤：

1. 安装 passwordcheck 扩展

首先，确保已安装 PostgreSQL 和 contrib 模块：

```
sudo yum install postgresql13-server postgresql13-contrib
```

根据你的 PostgreSQL 版本，可能需要调整 `postgresql13` 为你所使用的版本。

2. 启用 passwordcheck 扩展

将 `passwordcheck` 扩展添加到 PostgreSQL 配置文件 `postgresql.conf` 中：

```
sudo vi /var/lib/pgsql/13/data/postgresql.conf
```

添加以下行：

```
shared_preload_libraries = 'passwordcheck'
```

3. 配置密码复杂度策略

在 `postgresql.conf` 文件中添加具体的密码复杂度策略配置：

```
passwordcheck.min_length = 8
passwordcheck.max_length = 20
passwordcheck.min_lowercase = 1
passwordcheck.min_uppercase = 1
passwordcheck.min_digits = 1
passwordcheck.min_special = 1
```

保存并关闭文件。

4. 重启 PostgreSQL 服务

重启 PostgreSQL 以使配置生效：

```
sudo systemctl restart postgresql-13
```

5. 验证配置

尝试更改 PostgreSQL 用户密码，确保密码复杂度策略生效：

```
ALTER USER your_username WITH PASSWORD 'SimplePass'; -- 应该会失败
ALTER USER your_username WITH PASSWORD 'ComplexPass1!'; -- 应该会成功
```

注意事项

- **密码策略配置项**：`passwordcheck` 扩展中的配置项可能不如一些外部插件详细和可定制，但是它可以覆盖基本的密码复杂度要求。
- **数据库用户**：以上配置适用于 PostgreSQL 自身的数据库用户，而不是应用系统中的用户。

备份与恢复

使用copy语法备份与恢复

导出数据包含换行符的表

```
COPY my_table TO '/path/to/output_file.csv' WITH (FORMAT CSV, HEADER, QUOTE '"', ESCAPE '\');
```

导入包含换行符的数据

```
COPY my_table FROM '/path/to/output_file.csv' WITH (FORMAT CSV, HEADER, QUOTE '"', ESCAPE '\');
```

注意事项

- **CSV 格式**：确保你导出的 CSV 文件符合标准 CSV 格式，其中换行符、引号等都被正确转义。
- **QUOTE 和 ESCAPE 参数**：使用这些参数可以帮助正确处理包含换行符、引号、逗号等特殊字符的数据。