openGauss 安全体系创新

实践课



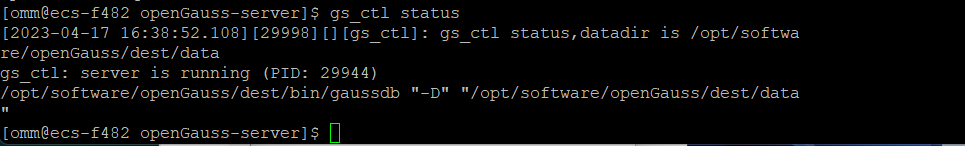
华为技术有限公司

# 关卡一、openGauss数据安装及基本操作

openGauss数据安装及基本操作, 作业提交任务如下：

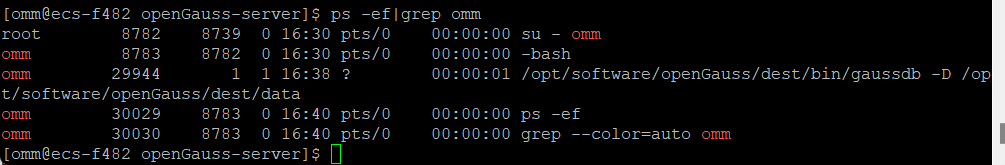
任务一：数据库状态验证

1. 查询数据库状态成功截图



任务二：数据库服务进程验证

1. 查看数据库服务进程截图（包含数据库服务器的主机名）



任务三：实践思考题

思考题1：为什么需要通过源码编译，安装数据库？

安装数据库是通过源码编译的原因主要是因为数据库软件通常包含多个模块，其中一些模块需要编译才能正常工作。此外，数据库软件的二进制文件通常是由编译器生成的，而不是直接由源代码编译生成的。因此，如果直接使用源代码编译，需要处理许多额外的细节，包括正确的库链接、配置和参数调整，以确保数据库软件能够正确编译和运行。

安装数据库的基本步骤如下:

* + - 1. 下载数据库源代码文件，通常是以.tar.gz 或.zip 等格式发布的。
      2. 解压源代码文件。
      3. 进入解压后的目录，运行 configure 命令，该命令将用于配置数据库软件的编译选项。
      4. 根据配置选项生成 makefile 文件。
      5. 运行 make 命令，编译数据库软件。
      6. 安装数据库软件，通常需要执行以下命令:

make install

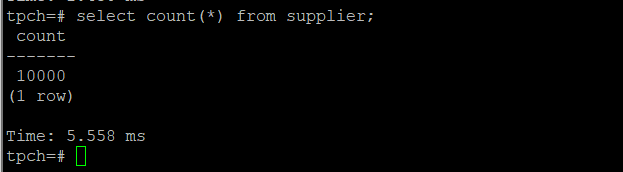
s

# 关卡二、openGauss数据导入及行存列存

任务一：数据初始化验证

1. 查询supplier表的行数，并将结果进行图：

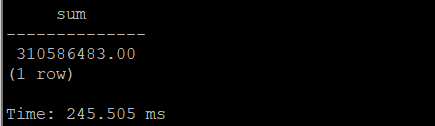
select count(\*) from supplier;;



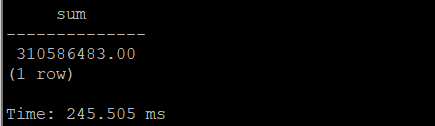
任务二：行存表与列存表执行效率对比

1. 2020年上半年litemall\_orders行存表与litemall\_orders\_col列存表中的order\_price的总和查询，并对比执行效率截图

select sum (order\_price) from litemall\_orders where add\_date between '20200101' and '20200701';



select sum (order\_price) from litemall\_orders\_col where add\_date between '20200101' and '20200701';



2. 2020年上半年litemall\_orders行存表与litemall\_orders\_col列存表中的order\_price的平均值查询，并对比执行效率截图

select avg (order\_price) from litemall\_orders where add\_date between '20200101' and '20200701';

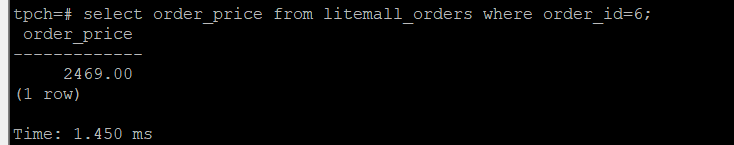


select avg (order\_price) from litemall\_orders\_col where add\_date between '20200101' and '20200701';

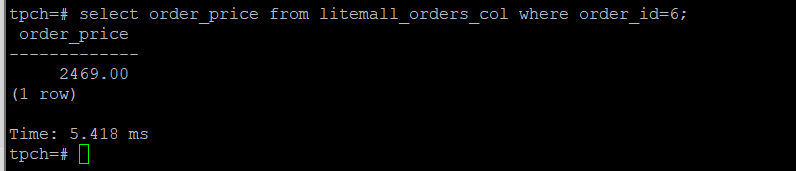


3. 查询litemall\_orders行存表与litemall\_orders\_col列存表中order\_id为6的order\_price的值，并对比执行效率截图。

select order\_price from litemall\_orders where order\_id=6;

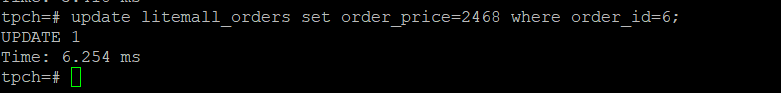


select order\_price from litemall\_orders\_col where order\_id=6;

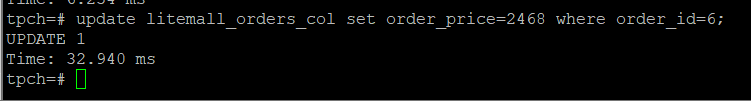


4. 将litemall\_orders行存表与litemall\_orders\_col列存表中order\_id为6的order\_price修改为2468，并对比执行效率截图。

update litemall\_orders set order\_price=2468 where order\_id=6;



update litemall\_orders\_col set order\_price=2468 where order\_id=6;



任务三：实践思考题

思考题1：

行存表与列存表在执行相同的SQL语句时，为何执行的时间不同？

行存表和列存表是两种常见的数据结构，分别用于解决不同类型的问题。行存表通常是用于处理大量单行数据，例如银行的账户余额表，每行数据代表一个账户，余额是该行数据的一个列。而列存表则是用于处理大量列数据，例如一篇文章包含多个关键词，每个关键词在文章中出现的次数是其中的一列。

思考题2：

在执行哪些类型SQL时，行存表效率更高？在执行哪些类型SQL时，列存表效率更高？

行存表更适合:

单行查询：行存表适合处理大量单行数据，因此如果 SQL 语句只需要查询单条记录，使用行存表会更高效。例如，如果要查询某个时间点上的某条记录，使用行存表可以更快地找到该记录。

批量插入：当要向表中插入大量数据时，行存表可能更加高效。因为行存表是按行存储数据的，所以可以在一行数据插入完毕后才将下一行数据插入，而不需要逐行插入。

索引查找：行存表的每个节点都存储了一行数据，因此如果 SQL 语句需要查找某一行数据，使用行存表可以更快地找到该行数据。

列存表更适合:

多行查询：如果 SQL 语句需要同时查询多行数据，列存表可能更加高效。因为列存表是按列存储数据的，所以可以在同时查询多行数据时，更快地定位需要查询的列。

数据更新：如果需要进行批量更新，列存表可能更加高效。因为列存表是按列存储数据的，所以可以更快地找到需要更新的列，并更新该列。

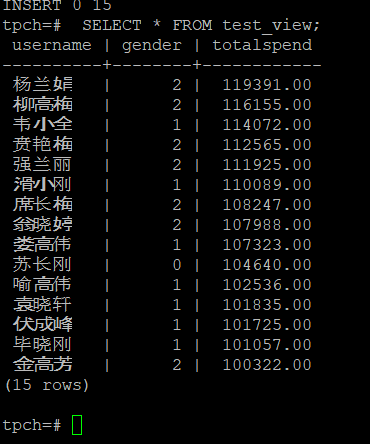
数据聚合：如果需要对数据进行聚合操作，列存表可能更加高效。因为列存表可以更快地对一列或多列进行计算，并将结果存储在相应的列中。

# 关卡三：openGauss物化视图应用

任务一：物化视图的使用

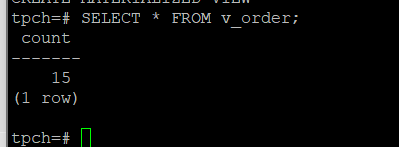
1. 创建物化视图所需要的表后，对表内容进行查询，对查询结果截图：

SELECT \* FROM test\_view;



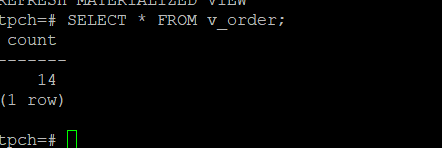
2. 使用物化视图统计人数，查询物化视图结果，将执行结果截图。

SELECT \* FROM v\_order;



3. 对表进行操作后，刷新物化视图，查询物化视图结果，将执行结果截图。

SELECT \* FROM v\_order;



4. 创建增量物化视图，查询物化视图结果，将执行结果截图。

SELECT \* FROM vi\_order;



5. 对表进行操作后，刷新增量物化视图，查询物化视图结果，将执行结果截图。

SELECT \* FROM vi\_order;



任务二：实践思考题

思考题1：全量物化视图与增量物化视图有哪些差别？

使用场景略有不同。下面是它们之间的主要差别:

数据更新方式：全量物化视图是基于全量数据更新的，即每次查询时都需要更新整个视图;而增量物化视图是基于增量数据更新的，即每次查询时只需要更新增量部分的数据。

视图数据量：全量物化视图的数据量相对于增量物化视图会更大，因为它需要更新整个视图的数据。而增量物化视图的数据量相对较小，因为它只更新增量部分的数据。

查询效率：在查询时，全量物化视图的效率更高，因为它可以直接访问视图中的全量数据。而增量物化视图的效率相对较低，因为它需要遍历整个视图，查找增量数据。

数据一致性：在全量物化视图中，数据的一致性得到保证，因为每次查询都会更新整个视图。而在增量物化视图中，数据的一致性可能会受到影响，因为每次查询只更新增量部分的数据。

应用场景：全量物化视图通常用于处理大量数据的快速查询，例如查询历史数据。而增量物化视图通常用于处理增量数据，例如在数据库中进行数据更新时，使用增量物化视图可以更快地更新数据。

思考题2：物化视图适用那些使用场景？

物化视图是一种用于优化查询的数据结构，它通常用于处理大量数据的快速查询和数据更新。下面是一些物化视图的适用场景:

数据仓库：在数据仓库中，通常需要对历史数据进行查询和分析。物化视图可以用于优化历史数据的查询效率，从而提高数据分析的效率。

数据挖掘：在数据挖掘中，通常需要对大量数据进行分组、聚合和计算。物化视图可以用于加速数据的查询和计算效率，从而提高数据挖掘的效率。

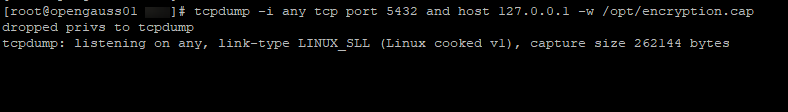
金融风控：在金融领域，通常需要对交易数据进行分析和监控，以保障金融安全。物化视图可以用于优化交易数据的查询效率，从而提高交易监控的效率。

数据可视化：在数据可视化中，通常需要对数据进行图表展示和分析。物化视图可以用于加速数据的查询和可视化效率，从而提高数据可视化的效果。

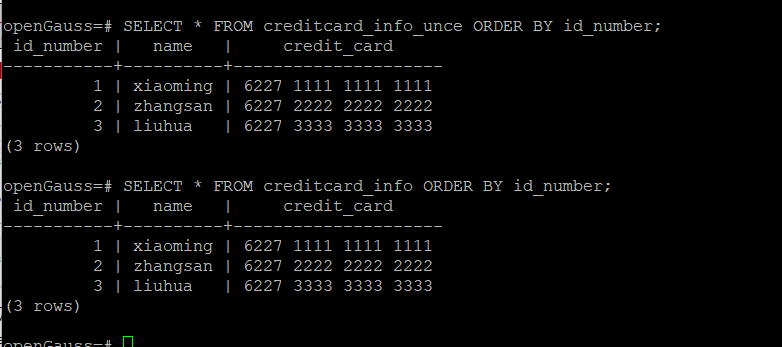
# 关卡四：openGauss密态数据库特性应用

任务一：物化视图的使用

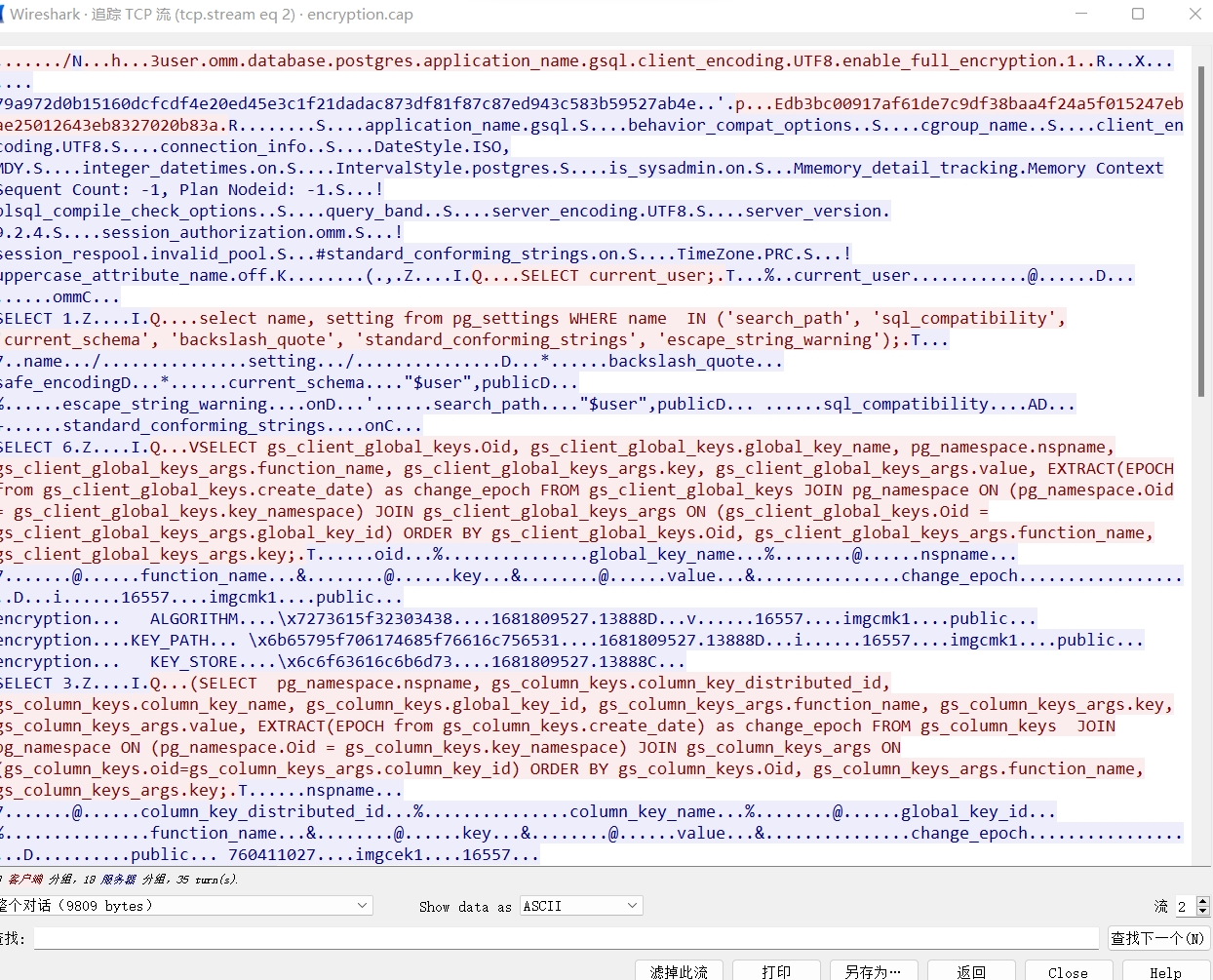
1. 通过tcpdump抓取数据流，此putty窗口暂时保持不动，将执行结果截图：



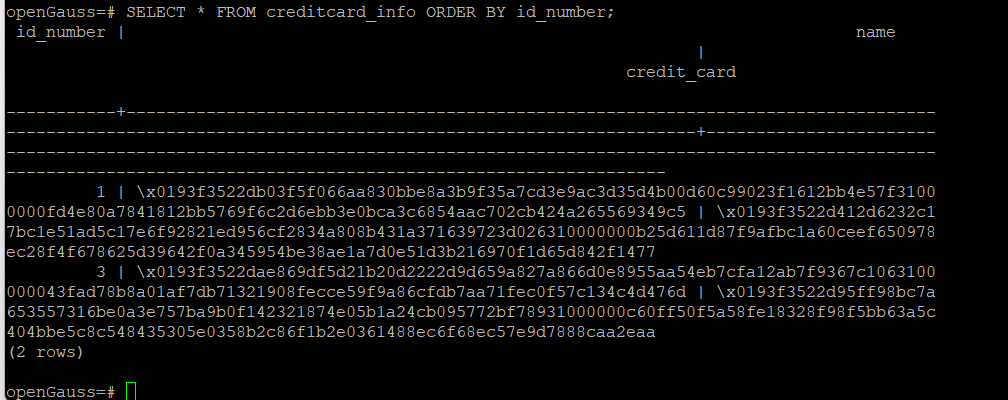
1. 将加密表和非加密表查询结果截图：



1. 用wireshark解析加密表和非加密表的差异时，非加密表name列和credit\_card列是明文，加密表name列和credit\_card列均是密文，将执行结果截图：



1. 查询加密表，查询到的结果为密文，将执行结果截图：



任务二：实践思考题

思考题1：

数据实际存储在物理磁盘上的时候是明文还是密文？数据的加解密的动作是在客户端完成的还是服务端完成的？

数据在物理磁盘上存储的时候通常是明文，即数据不加解密的状态。这是因为在磁盘上存储数据时，数据通常是被操作系统和文件系统进行处理和存储的，而操作系统和文件系统通常不支持对数据进行加密。因此，数据在磁盘上是明文存储的。

数据的加解密动作通常是在服务端完成的。在客户端和服务器之间传输的数据通常是明文，因此如果客户端和服务器之间需要进行数据加密，通常需要在客户端和服务器之间建立加密通道，或者在数据发送和接收时使用加密算法进行加密和解密。

需要注意的是，有些数据库和文件系统支持对数据进行加密存储。在这种情况下，数据的加解密动作可以在客户端和服务器之间进行，也可以在数据库和文件系统之间进行。但需要注意的是，加密会增加数据访问和处理的时间和成本，因此在实际应用中需要根据具体情况来选择合适的加密方案和加密方式。