openGauss 安全体系创新

实践课



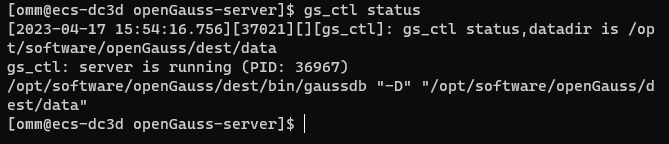
华为技术有限公司

# 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？

定制化需求：通过源码编译，用户可以根据自己的需求对数据库进行定制，例如修改默认设置、添加特定的功能或优化性能。

兼容性：编译源码可以确保在特定操作系统和硬件平台上获得良好的兼容性，特别是在某些非主流平台上。编译过程可以自动检测和适应系统环境，解决潜在的兼容性问题。

更新和补丁：源码编译安装允许用户应用最新的补丁和更新。这有助于在官方发布新版本之前获得修复程序、安全更新或新功能。

性能优化：用户可以根据自己的硬件和应用需求调整编译选项，以获得更好的性能。例如，可以启用特定的CPU指令集或优化内存使用等。

学习和研究：通过阅读和编译源码，开发者可以更深入地了解数据库的工作原理和内部结构，从而提高自己的技能和知识。

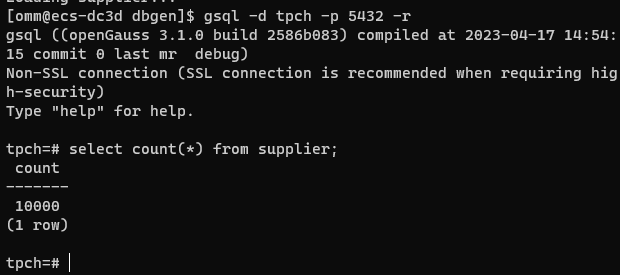
总的来说，通过源码编译安装数据库可以让用户获得更高的灵活性和定制化能力，但也可能带来更多的维护和配置工作。对于大多数用户来说，使用预编译的二进制包或者使用包管理器进行安装可能是更为简单和方便的选择。

# 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：

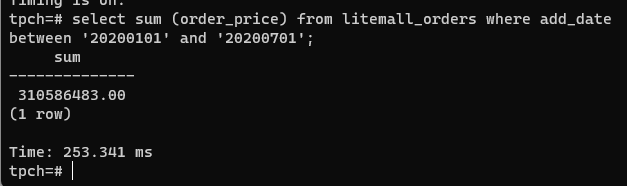
select count(\*) from supplier;;



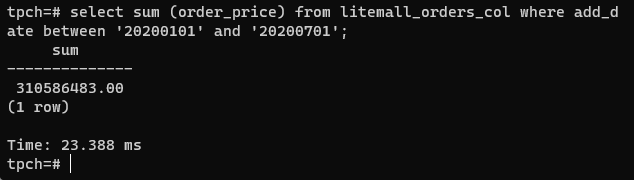
任务二：行存表与列存表执行效率对比

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';

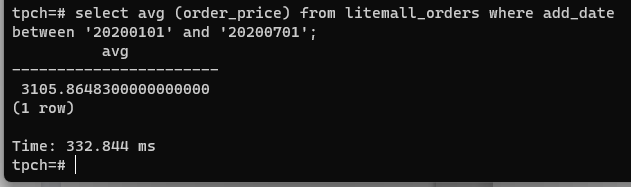


select sum (order\_price) from litemall\_orders\_col where add\_date between '20200101' and '20200701';

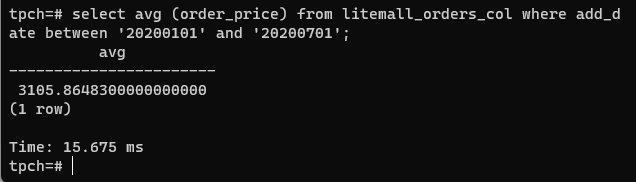


2. 2020年上半年litemall\_orders行存表与litemall\_orders\_col列存表中的order\_price的平均值查询，并对比执行效率截图

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

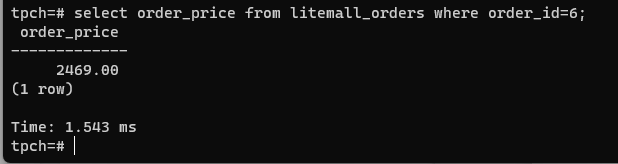


select avg (order\_price) from litemall\_orders\_col where add\_date between '20200101' and '20200701';

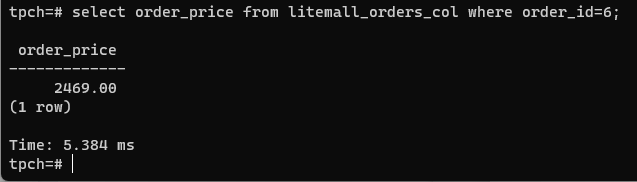


3. 查询litemall\_orders行存表与litemall\_orders\_col列存表中order\_id为6的order\_price的值，并对比执行效率截图。

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

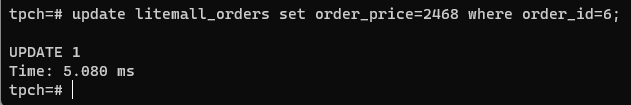


select order\_price from litemall\_orders\_col where order\_id=6;

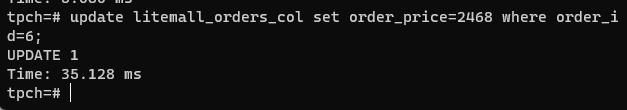


4. 将litemall\_orders行存表与litemall\_orders\_col列存表中order\_id为6的order\_price修改为2468，并对比执行效率截图。

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？

数据读取：列存表在只访问部分列的查询中需要读取的数据量更少，因此执行时间更短。而行存表在这种情况下需要读取整行数据，导致执行时间更长。

数据压缩：由于列存表中相邻数据的相似性更高，因此数据压缩效果更好。这可以减少磁盘I/O操作，从而缩短执行时间。而行存表中数据的相似性较低，压缩效果相对较差。

缓存利用：列存表可以更好地利用CPU缓存，因为只需要将所需的列数据加载到缓存中。而行存表需要将整行数据加载到缓存中，这可能导致缓存的空间利用率降低。

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？

行存表效率更高的SQL类型：

事务性操作：行存表在处理事务性操作（如插入、更新和删除）时性能较好。这是因为事务性操作通常涉及整行数据的访问和处理，而行存表将整行数据存储在一起，可以快速访问和处理。

多列数据访问：当SQL查询需要访问和处理多列数据时，行存表往往具有更好的性能。例如，当查询需要返回大量列或者涉及多表连接时，行存表可以快速访问整行数据，而列存表需要从多个列中提取数据并重新组合成行，这会增加处理时间。

列存表效率更高的SQL类型：

聚合查询：列存表在处理聚合查询（如 COUNT、SUM、AVG、MIN 和 MAX 等）时性能较好。这是因为聚合查询通常只涉及少数几列数据，而列存表可以只读取这些列数据，从而减少不必要的数据读取。

分析性查询：对于分析性查询和报表生成等场景，列存表往往具有更好的性能。这些查询通常只涉及部分列数据，并且可能涉及大量的数据过滤、排序和分组操作。列存表可以只读取所需列数据，并且由于相邻数据的相似性更高，可以更有效地进行数据压缩，从而减少磁盘I/O操作。

列过滤：当SQL查询需要基于某些列进行数据过滤时，列存表具有更好的性能。例如，当查询使用 WHERE 子句对某个列进行筛选时，列存表可以只读取该列数据，减少数据读取量，而行存表需要读取整行数据。

# 关卡三：openGauss物化视图应用

任务一：物化视图的使用

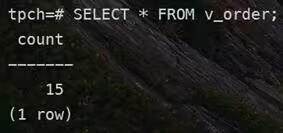
1. 创建物化视图所需要的表后，对表内容进行查询，对查询结果截图：

SELECT \* FROM test\_view;



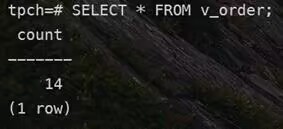
2. 使用物化视图统计人数，查询物化视图结果，将执行结果截图。

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？

更新策略：

全量物化视图：在全量物化视图中，每次更新时，系统会重新计算整个物化视图，不论基表数据的变化大小。这可能导致更新全量物化视图的过程耗费较多时间和资源。

增量物化视图：相较于全量物化视图，增量物化视图只针对基表中发生变化的数据进行更新。这种更新策略通常更加高效，因为它仅处理基表的增量变化，而非重新计算整个物化视图。

性能：

全量物化视图：由于需要重新计算整个物化视图，全量物化视图的更新过程可能会消耗大量时间和系统资源。在数据量较大的情况下，这种更新策略可能导致性能瓶颈。然而，在某些场景下，全量物化视图可能适用，例如对实时性要求不高、数据变动较小或更新频率较低的场景。

增量物化视图：增量物化视图通常具有更好的性能，因为它仅处理基表的增量变化。在需要频繁更新物化视图的场景下，增量物化视图是更为高效的选择。但是，创建增量物化视图的过程可能较为复杂，因为需要构建适当的更新机制来识别基表的增量变化。

思考题2：物化视图适用那些使用场景？

数据仓库与决策支持系统（DSS）：物化视图通常用于数据仓库和决策支持系统，因为这些系统中经常涉及复杂的聚合操作和跨表查询。物化视图可以预先计算这些操作的结果，提高查询性能。

数据报表和汇总：物化视图适用于那些需要定期生成报表或汇总数据的场景。通过预先计算汇总信息并存储在物化视图中，可以加快报表生成速度，减轻数据库服务器的压力。

数据分析和挖掘：物化视图可以用于存储经常用于数据分析和挖掘的查询结果。这可以避免每次分析时都重新执行相同的查询，提高数据分析速度。

数据复制和分布式系统：物化视图可以用于在分布式系统中同步数据。通过在远程站点创建物化视图，可以实现数据的实时或定期同步，降低网络带宽消耗。

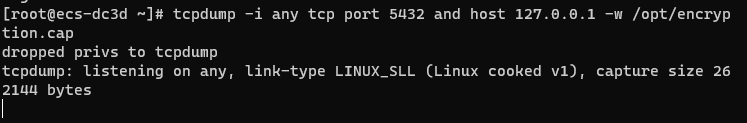
性能优化：物化视图可以用于优化查询性能。在面临复杂查询和大量数据时，物化视图可以通过预先计算和存储查询结果，降低实时查询的计算和I/O负担。

数据历史和审计：物化视图可以用于存储数据的历史快照。通过定期刷新物化视图，可以追踪数据的变化，便于进行数据审计和趋势分析。

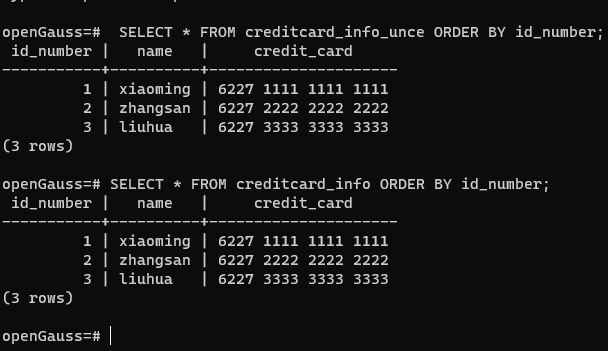
# 关卡四：openGauss密态数据库特性应用

任务一：物化视图的使用

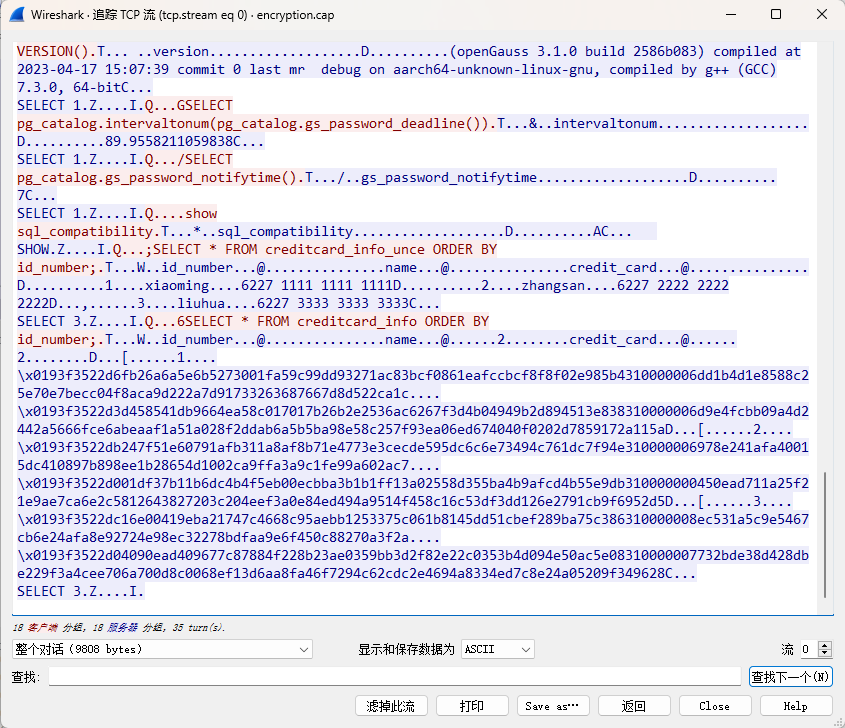
1. 通过tcpdump抓取数据流，此putty窗口暂时保持不动，将执行结果截图：



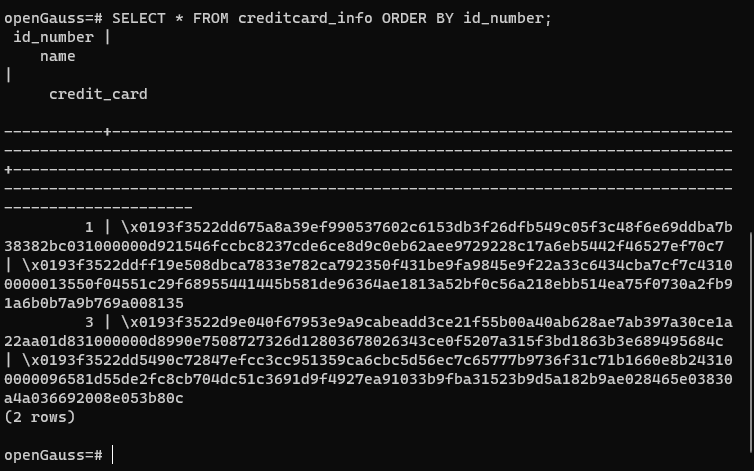
1. 将加密表和非加密表查询结果截图：



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？

密文。

在客户端完成。