openGauss 安全体系创新

实践课



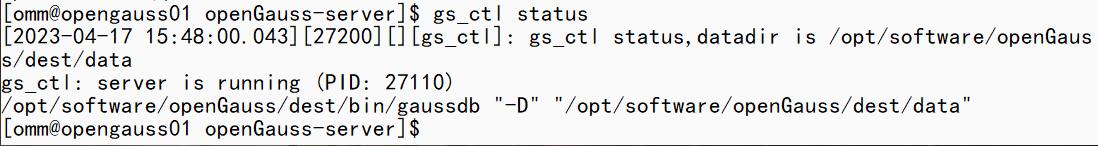
华为技术有限公司

# 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：

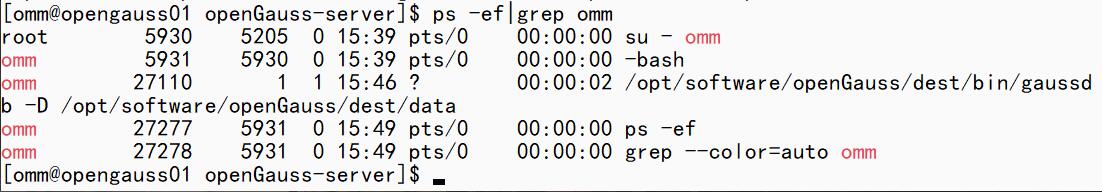
任务一：数据库状态验证

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？

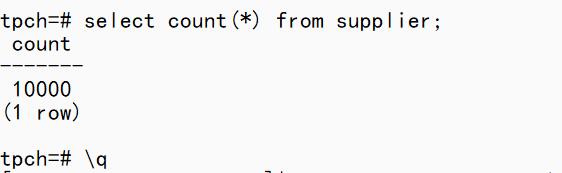
1. **定制化：通过源码编译安装，可以根据自己的需要进行定制化配置，包括指定特定的编译选项、安装目录、模块等。**
2. **最新版本：源码编译安装可以确保你安装的是最新的版本，而不是依赖于操作系统提供的版本。这样可以获得最新的功能和修复的 bug。**
3. **最大性能：源码编译可以根据硬件环境和需要进行优化，以获得最大性能。而预编译的二进制版本则可能无法完全利用硬件性能。**
4. **安全性：通过源码编译安装数据库，可以避免使用可能存在漏洞或被恶意修改的预编译版本，提高安全性。**
5. **学习机会：通过源码编译安装，可以深入了解数据库的结构和运作原理，从而提高对数据库的理解和掌握能力。**

# 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：

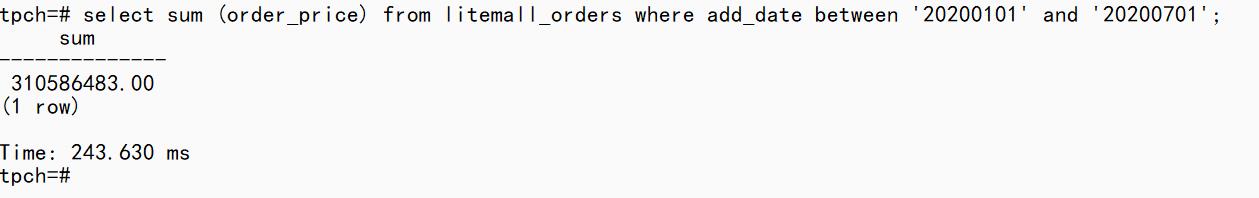
select count(\*) from supplier;;



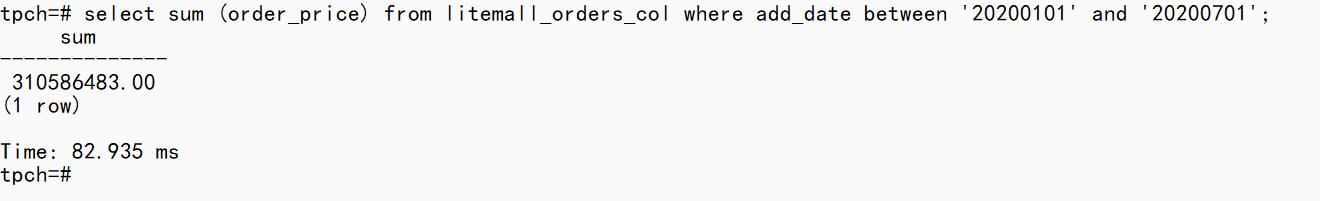
任务二：行存表与列存表执行效率对比

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';

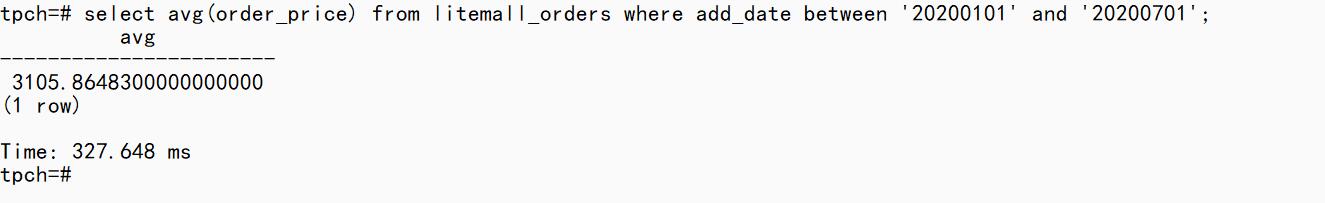


select sum (order\_price) from litemall\_orders\_col where add\_date between '20200101' and '20200701';

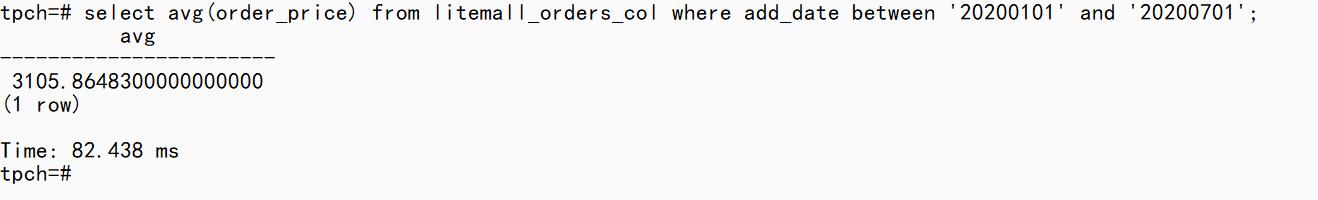


2. 2020年上半年litemall\_orders行存表与litemall\_orders\_col列存表中的order\_price的平均值查询，并对比执行效率截图

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

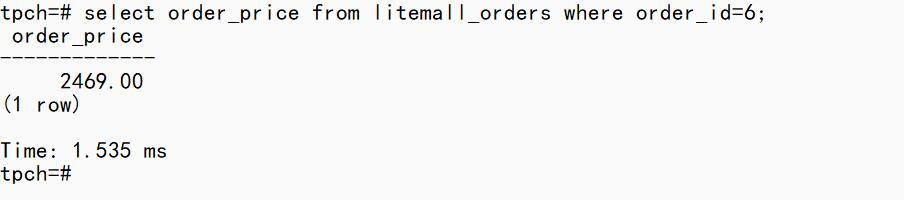


select avg (order\_price) from litemall\_orders\_col where add\_date between '20200101' and '20200701';

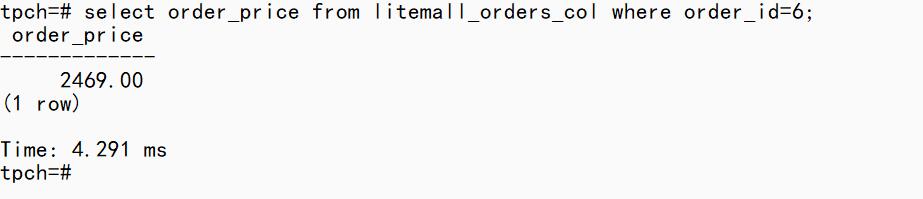


3. 查询litemall\_orders行存表与litemall\_orders\_col列存表中order\_id为6的order\_price的值，并对比执行效率截图。

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

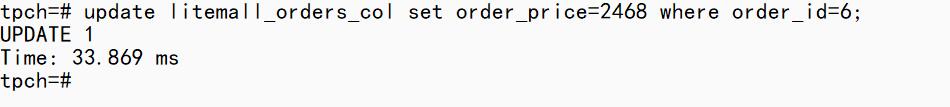


select order\_price from litemall\_orders\_col where order\_id=6;

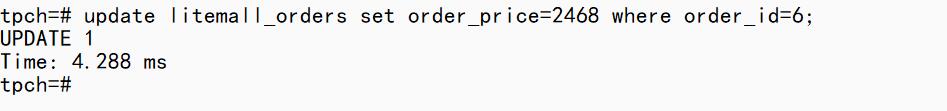


4. 将litemall\_orders行存表与litemall\_orders\_col列存表中order\_id为6的order\_price修改为2468，并对比执行效率截图。

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？

**行存表和列存表的物理存储结构不同，在行存表中数据是按照行的方式存储的，在列存表中，数据是按照列的方式存储的。**

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？

**行存表效率更高的情况：**

1. **小规模的数据：当数据量较小且查询结果需要多个列时，行存表的查询效率更高，因为列存表需要访问更多的列。**
2. **插入、更新或删除操作：行存表在插入、更新或删除数据时，只需要操作一整行的数据，而列存表则需要对多个列进行操作，因此行存表的效率更高。**
3. **多表关联查询：当需要进行多表关联查询时，行存表可能会比列存表更快，因为行存表可以利用索引来加速查询。**

**列存表效率更高的情况：**

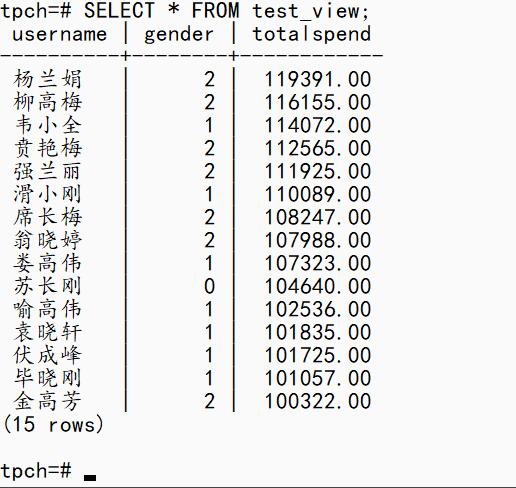
1. **大规模数据查询：当数据量非常大时，列存表的查询效率更高，因为只需要读取需要的列，可以大大减少读取数据的量，从而提高查询速度。**
2. **聚合查询：当需要进行聚合查询时，列存表通常比行存表更快，因为聚合查询通常只需要读取几列数据，而列存表可以更快地访问这些列。**
3. **大批量的数据导入或导出：列存表的批量插入、导入、导出等操作比行存表更高效，因为列存表可以快速地按列操作。**

# 关卡三：openGauss物化视图应用

任务一：物化视图的使用

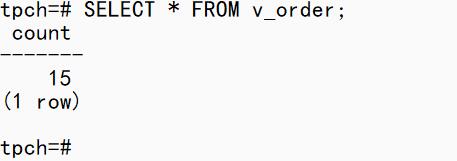
1. 创建物化视图所需要的表后，对表内容进行查询，对查询结果截图：

SELECT \* FROM test\_view;



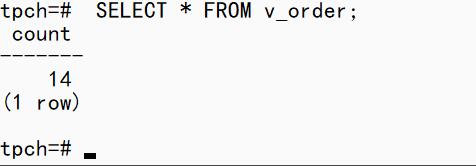
2. 使用物化视图统计人数，查询物化视图结果，将执行结果截图。

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;



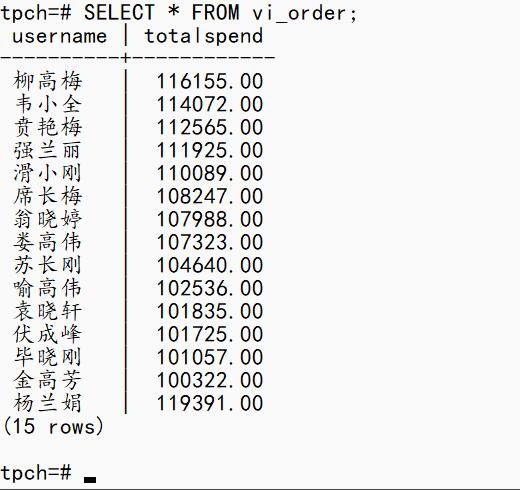
4. 创建增量物化视图，查询物化视图结果，将执行结果截图。

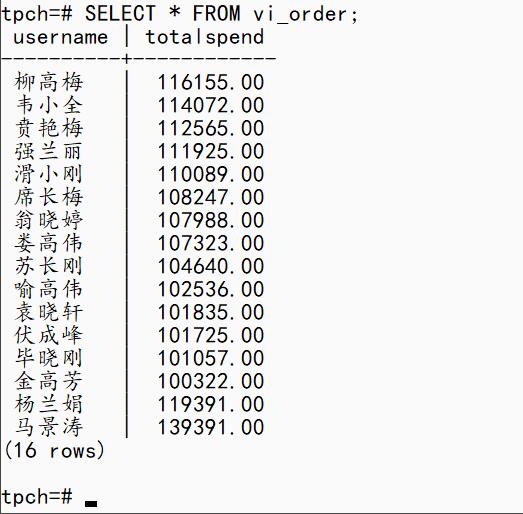
SELECT \* FROM vi\_order;



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？

1. **刷新方式不同：全量物化视图在刷新时需要重新计算整个视图的数据，而增量物化视图只需要计算新增、修改和删除的数据，因此在数据量较大的情况下，增量物化视图刷新效率更高。**
2. **使用场景不同：全量物化视图适用于数据变化比较频繁但数据量较小的情况，而增量物化视图适用于数据变化比较少但数据量较大的情况。**
3. **内存占用不同：全量物化视图需要占用较多的内存空间来存储视图的数据，而增量物化视图只需要存储新增、修改和删除的数据。**

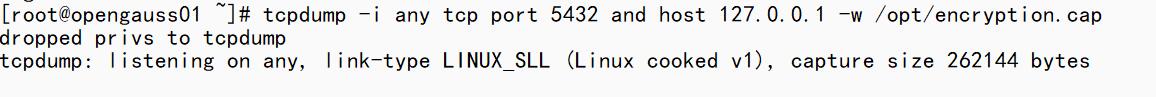
思考题2：物化视图适用那些使用场景？

1. **大量数据的复杂查询：当数据量非常大时，复杂查询可能需要花费较长的时间才能返回结果，因此可以使用物化视图来提前计算和缓存查询结果，从而加速查询。**
2. **频繁查询的数据：当数据需要频繁查询时，可以使用物化视图来提高查询性能，减少重复计算的时间。**
3. **分析型查询：当需要进行复杂的分析型查询时，可以使用物化视图来预先计算和存储数据，从而加速查询。**

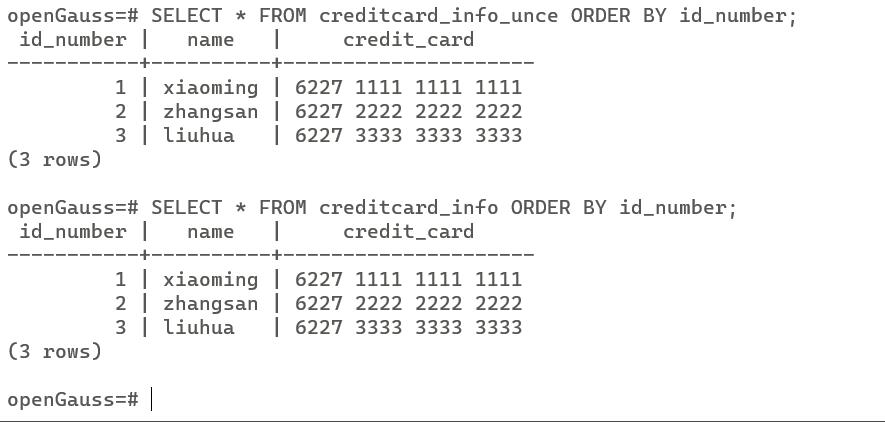
# 关卡四：openGauss密态数据库特性应用

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？

**一般情况下，数据都是以明文存储在物理磁盘上，数据的加解密动作在客户端完成。**