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：

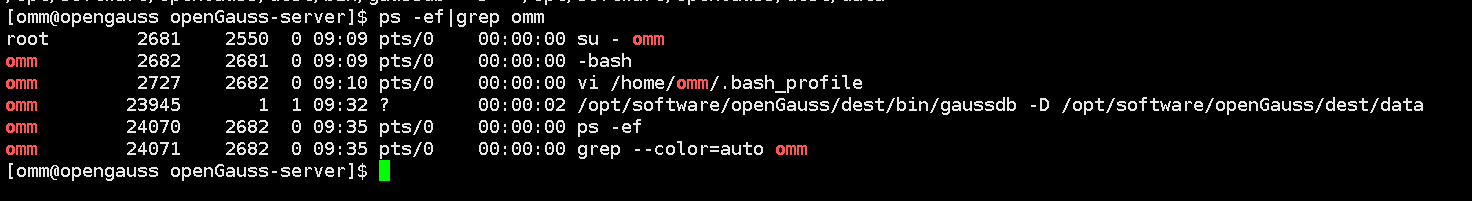
任务一：数据库状态验证

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？

通过源码编译和安装数据库有以下几个优点：

定制化：通过源码编译，可以对数据库进行各种参数配置，以适应不同的应用场景，例如可以为某个特定的应用程序定制一个数据库版本，以满足其性能和功能需求。

最新版本：通过源码编译，可以安装最新版本的数据库，以获取最新的功能和性能优化，而不受发行版更新速度的限制。

安全性：通过源码编译，可以确保软件是从可信源代码构建的，减少了潜在的安全隐患。

可控性：通过源码编译，可以直接控制编译和安装过程，以确保软件在系统中的运行环境和依赖项等方面满足特定的要求和期望。

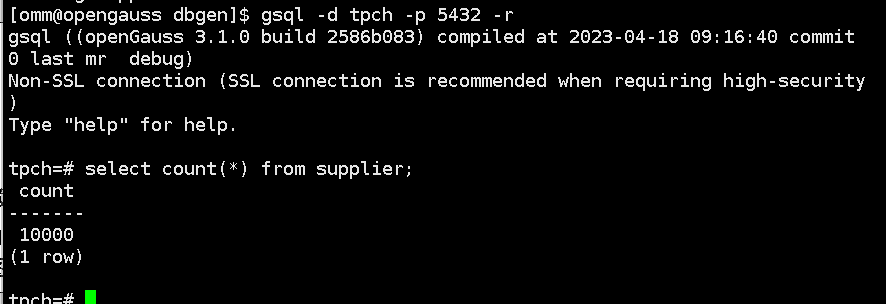
教育性：通过源码编译，可以更好地了解数据库的内部工作原理和实现细节，有助于学习数据库相关的知识和技能。

# 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：

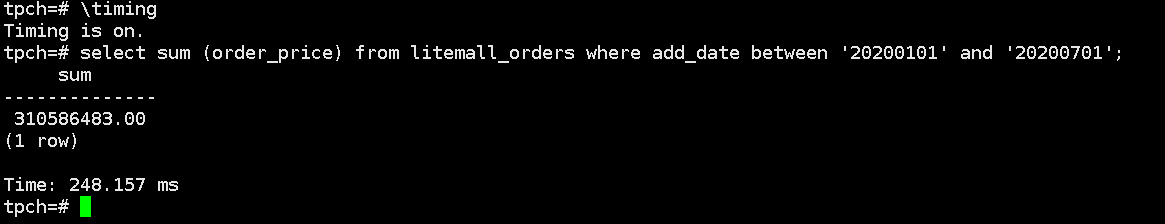
select count(\*) from supplier;;



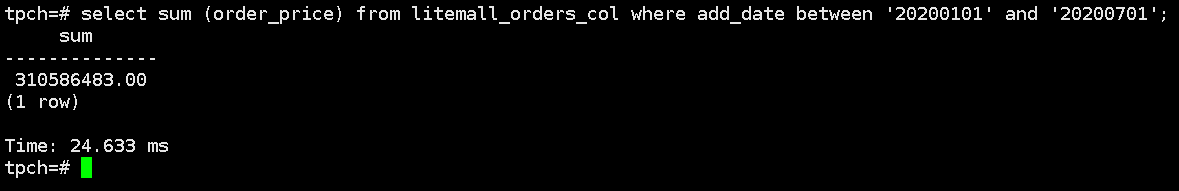
任务二：行存表与列存表执行效率对比

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';

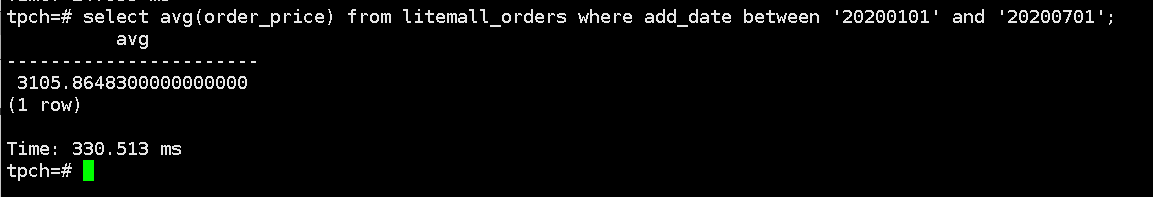


select sum (order\_price) from litemall\_orders\_col where add\_date between '20200101' and '20200701';

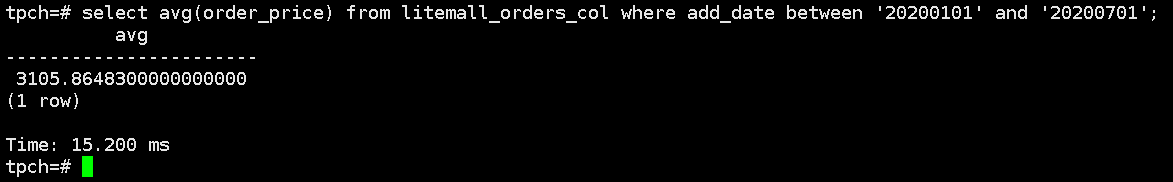


2. 2020年上半年litemall\_orders行存表与litemall\_orders\_col列存表中的order\_price的平均值查询，并对比执行效率截图

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

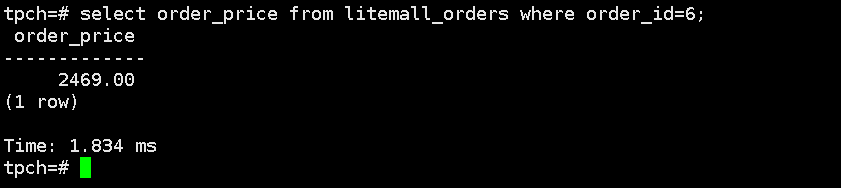


select avg (order\_price) from litemall\_orders\_col where add\_date between '20200101' and '20200701';

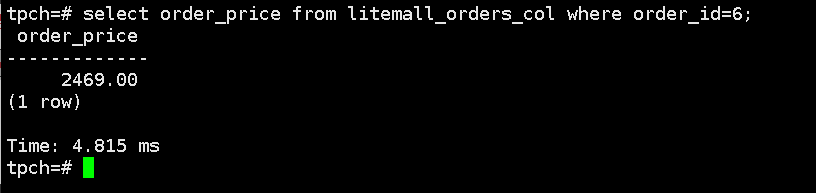


3. 查询litemall\_orders行存表与litemall\_orders\_col列存表中order\_id为6的order\_price的值，并对比执行效率截图。

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

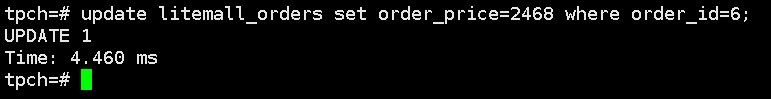


select order\_price from litemall\_orders\_col where order\_id=6;

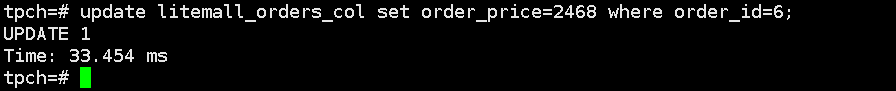


4. 将litemall\_orders行存表与litemall\_orders\_col列存表中order\_id为6的order\_price修改为2468，并对比执行效率截图。

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不同的数据库存储方式。行存表按照行来存储数据，而列存表按照列来存储数据。

当执行相同的SQL语句时，执行时间不同的原因是由于它们的物理存储结构不同，因此在访问和处理数据时的方式也不同。具体来说，以下是两种存储方式的特点及其影响：

行存表

行存表的特点是每行数据都存储在一起。当执行SQL语句时，需要读取整行数据，包括不需要的列，因此查询的速度较慢，尤其是对于大量的列和行时，效率会更低。此外，由于需要对整行数据进行加锁和释放锁，因此在高并发的情况下，容易造成阻塞和性能瓶颈。

列存表

列存表的特点是将同一列的数据存储在一起。当执行SQL语句时，只需要读取需要的列，而不需要读取整行数据，因此查询速度较快，尤其是在需要查询大量列的情况下。此外，由于只需要对需要的列进行加锁和释放锁，因此在高并发的情况下，性能更好。

综上所述，行存表和列存表在执行相同的SQL语句时，执行时间不同的原因是由于它们的物理存储结构和数据访问方式不同所致。在实际使用中，应根据具体的需求和情况选择合适的存储方式。

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？

一般来说，行存表和列存表的性能各有优劣，因此需要根据具体的SQL类型和数据特点来选择合适的存储方式。以下是一些一般情况下，行存表和列存表分别适合的SQL类型：

适合行存表的SQL类型：

基于主键或唯一键的查询：由于行存表将一行数据存储在一起，因此在查询主键或唯一键时，可以直接通过索引快速定位到对应的行，因此查询效率较高。

需要查询少量列或需要全表扫描的查询：由于行存表在查询时需要读取整行数据，因此在需要查询少量列或需要全表扫描的查询时，行存表的效率可能更高。

需要进行大量的更新和插入操作：由于行存表将一行数据存储在一起，因此在进行大量的更新和插入操作时，行存表的效率可能更高。

适合列存表的SQL类型：

需要查询大量列的查询：由于列存表将同一列的数据存储在一起，因此在需要查询大量列的查询时，列存表的效率可能更高。

需要进行聚合计算的查询：由于列存表将同一列的数据存储在一起，因此在进行聚合计算的查询时，列存表的效率可能更高。

需要进行少量更新和插入操作的查询：由于列存表将同一列的数据存储在一起，因此在进行少量更新和插入操作的查询时，列存表的效率可能更高。

需要注意的是，上述的适合行存表和列存表的SQL类型仅是一般情况下的建议，实际的应用场景可能更加复杂，需要根据具体的需求和数据特点进行选择。

# 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;



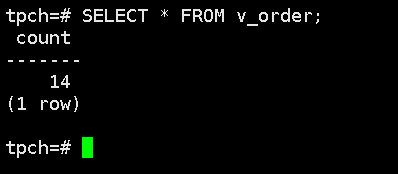
2. 使用物化视图统计人数，查询物化视图结果，将执行结果截图。

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？

全量物化视图和增量物化视图都是用于提高查询性能的技术，但它们的刷新方式和使用场景略有不同。

全量物化视图是指将查询结果缓存到一个物化视图中，这个物化视图中包含了所有的结果数据。当原始表数据发生改变时，需要手动或自动地刷新物化视图，以便物化视图中的数据与原始表中的数据同步。全量物化视图适用于查询结果数据不频繁变化的场景，如每天生成的报表等。

增量物化视图是指将查询结果缓存到一个物化视图中，但这个物化视图只包含了部分结果数据。当原始表数据发生改变时，只需要更新物化视图中与改变数据相关的部分数据，而不是全部数据。增量物化视图适用于查询结果数据变化频繁的场景，如在线分析应用（OLAP）等。

总之，全量物化视图和增量物化视图都是用于提高查询性能的技术，但它们的刷新方式和使用场景不同。

思考题2：物化视图适用那些使用场景？

物化视图（Materialized Views）适用于需要频繁查询大量数据的情况，可以通过将查询的结果预先计算并保存在物化视图中来加快查询速度。

以下是一些适合使用物化视图的场景：

大型数据仓库：物化视图可以提高数据仓库中查询的性能，尤其是对于复杂查询，它可以将查询结果缓存起来，减少查询所需的时间和资源。

复杂查询：当查询需要执行多个连接、聚合和计算时，使用物化视图可以减少查询的执行时间和资源消耗。

网络延迟高的环境：在网络延迟较高的情况下，使用物化视图可以减少查询的响应时间，提高应用程序的性能。

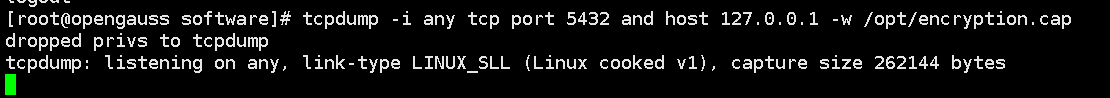
可以承受数据不一致的场景：物化视图是一种预先计算的结果，当底层数据发生变化时，物化视图中的数据不会立即更新，这意味着物化视图可能会与底层数据不一致。因此，适合使用物化视图的场景是可以承受一定程度的数据不一致的场景，例如报告和分析领域。

需要注意的是，使用物化视图可能会增加系统的存储需求，因为物化视图需要存储计算结果。因此，在使用物化视图时需要权衡存储和查询性能之间的平衡。

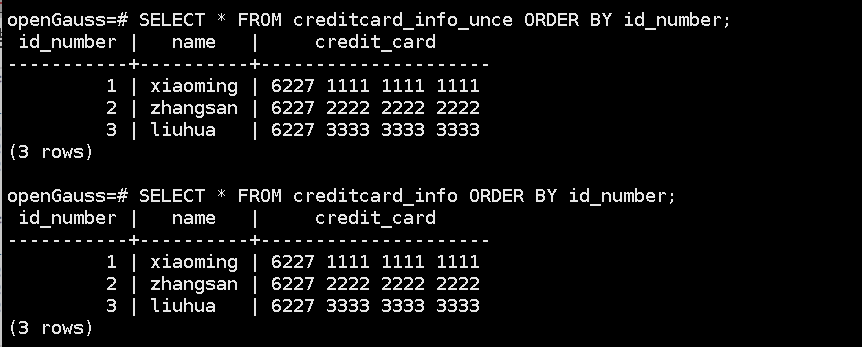
# 关卡四：openGauss密态数据库特性应用

任务一：物化视图的使用

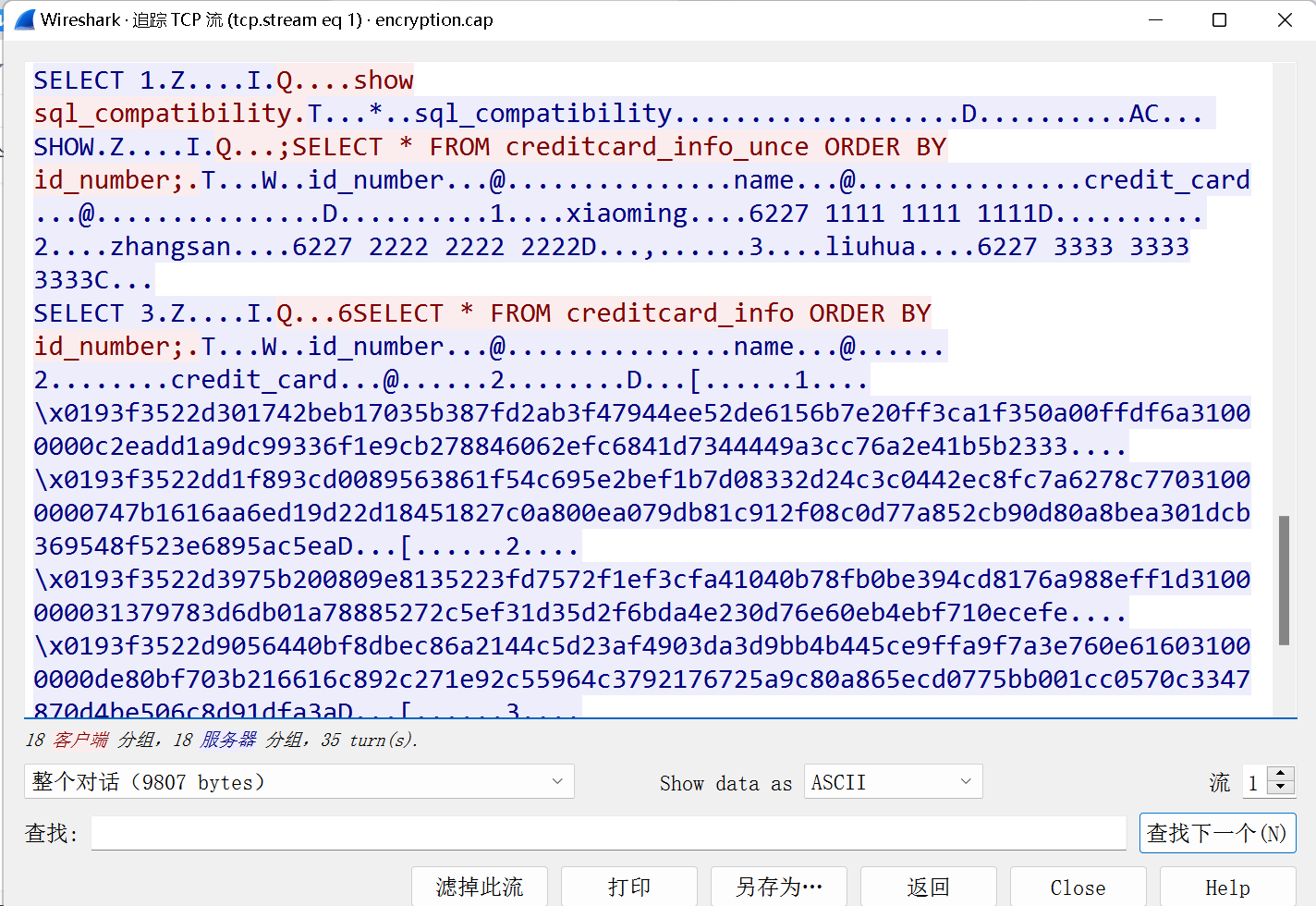
1. 通过tcpdump抓取数据流，此putty窗口暂时保持不动，将执行结果截图：



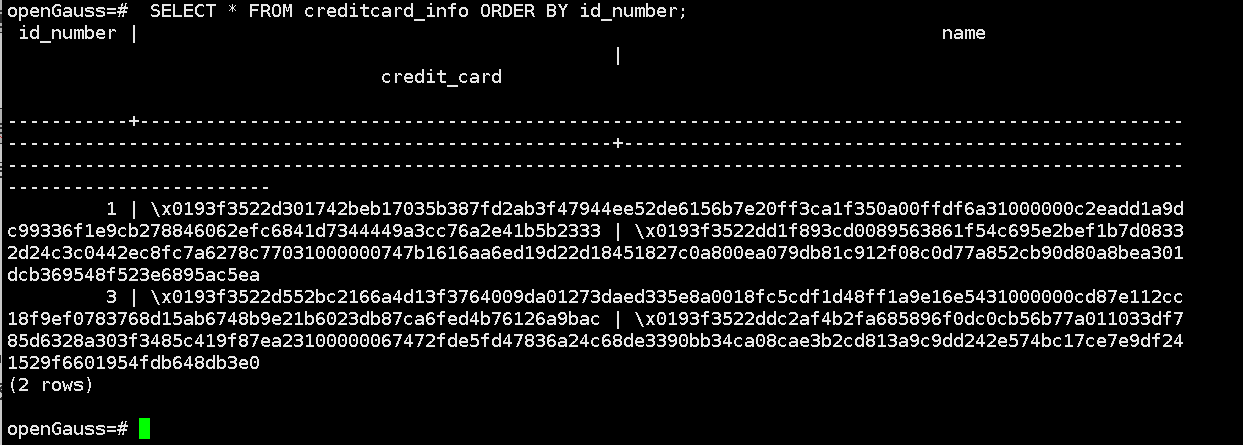
1. 将加密表和非加密表查询结果截图：



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？

数据实际存储在物理磁盘上时通常是密文形式，以保护数据的安全性。这通常涉及将原始数据使用加密算法进行加密处理，然后将加密后的数据存储在磁盘上。

加解密的动作可以在客户端或服务端完成，具体取决于系统的设计。在某些情况下，客户端可能会在本地加密数据，然后将加密后的数据上传到服务端进行存储。在其他情况下，服务端可能会负责加密和解密数据，以保护数据的安全性。

无论是客户端还是服务端进行加解密，都需要确保安全性和保密性。例如，如果加密密钥在传输过程中被泄露或遭到攻击，数据的保密性可能会受到威胁。因此，加密和解密的实现需要依靠安全的算法和协议，并且需要对数据进行适当的保护和控制访问权限。