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

图形用户界面, 文本

描述已自动生成

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）

文本

描述已自动生成

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？

答：

1：满足不同的运行平台，我们Linux发型版本众多，但是每个版本采用的软件或者内核版本都不一样，而我们的二进制包所依赖的环境不一定能够正常运行，所以需要源码来保证能够正常运行；

2：方便定制，满足不同的需求，很多时候我们所需要的软件都是可以定制的，我需要什么就安装什么，大多数二进制代码都是一键装全，所以自由度并不高；

3：方便运维、开发人员维护，我们的源码是可以打包二进制的，如果是对整个软件打包，代价不小，并且维护较难，但如果是源码的话，软件产商会直接维护，并且是二进制，一般都是Linux发行商提供。

# 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;;

文本

描述已自动生成

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';

文本

描述已自动生成

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

文本

描述已自动生成

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';

文本

描述已自动生成

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

文本

描述已自动生成

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;

文本

描述已自动生成

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

文本

描述已自动生成

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;

文本

描述已自动生成

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

文本

描述已自动生成

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？

答：

因为行存储的结构比较固定，每一行都必须一样，尽管不用也必须空在那里，这样就会占用不少存储空间，并且在执行SQL语句时也需要扫描，造成时间效率低；而列存储是非关系型数据库，倾向于结构弱化，按需存储，不会造成浪费，所以查询时也较短。

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？

答：

行存表主要适用于：关注整张表内容，或者需要经常更新数据、需要经常读取整行数据、不需要聚集运算，或者快速查询需求、数据表本身数据行并不多

列存表主要适用于：经常关注一张表某几列而非整表数据的时候、基于一列或比较少的列计算的时候、数据表拥有非常多的列的时候、数据表有非常多行数据并且需要聚集运算的时候、数据表列里有非常多的重复数据，有利于高度压缩

# 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;

图片包含 图示

描述已自动生成

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;

文本

描述已自动生成

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;

文本

描述已自动生成

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;

图形用户界面, 文本

中度可信度描述已自动生成

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;

图形用户界面

中度可信度描述已自动生成

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？

答：

全量物化视图支持与CREATE TABLE AS语句一致；创建索引；支持analyze、explain，不支持增删改操作，只支持查询语句。

增量物化视图支持单表查询语句、多个单表查询的UNION ALL、物化视图上支持创建索引和Analyze操作，不支持多表join连接计划及subquery计划，除少部分ALTER操作外，不支持基表执行绝大多数DDL操作，不支持增删改操作，不支持临时表/hashbucket/unlog/分区表创建视图物化视图，不支持物化视图嵌套创建，仅支持行存表，不支持UNLOGGED类型的物化视图，不支持WITH语法。

思考题2：物化视图适用那些使用场景？

答：

1. 单表查询语句。

2. 多个单表查询的UNION ALL。

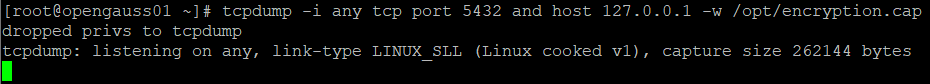
3. 物化视图上支持创建索引。

4. 物化视图支持Analyze操作。

# 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：

日历

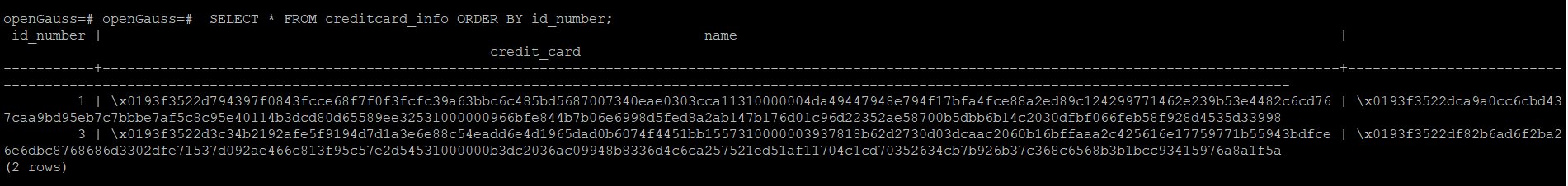
描述已自动生成

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：

文本

描述已自动生成

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？

答：数据实际存储在物理磁盘上的时候是密文。加密是在服务器端完成的，而解密是在客户端完成的。