关卡2-2

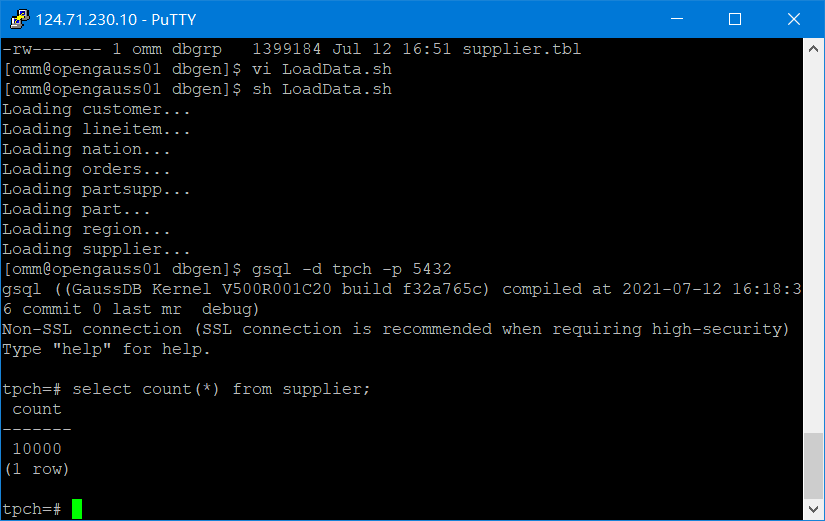
openGauss数据导入及基本操作

openGauss数据导入及基本操作

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：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。

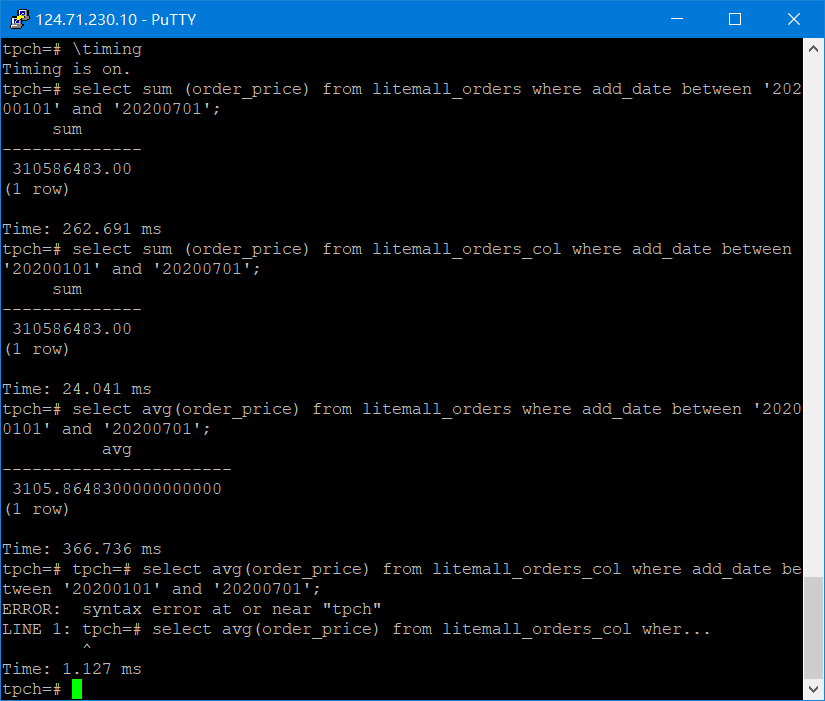
SELECT \* FROM v\_order;

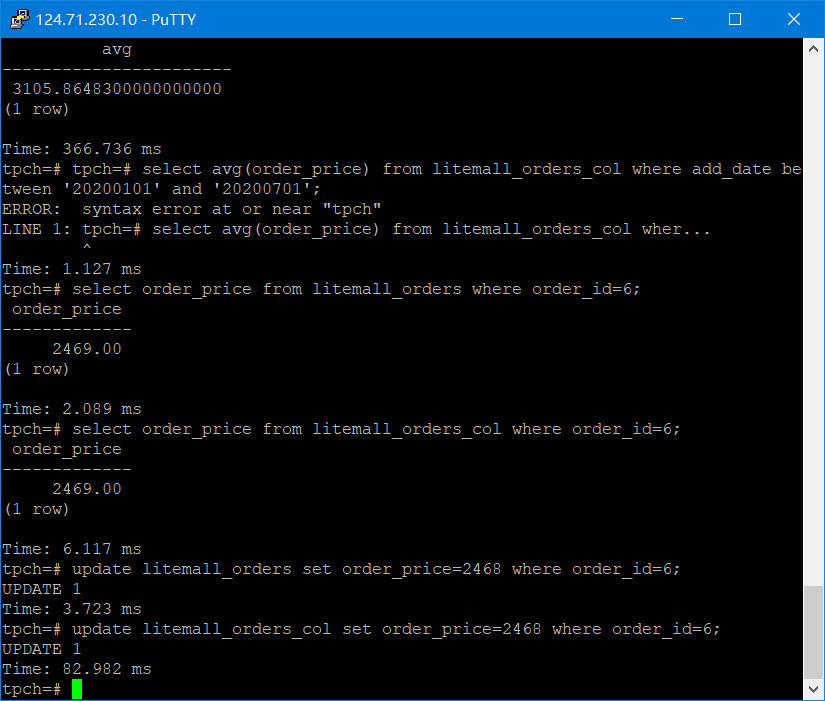
4. 创建增量物化视图，查询物化视图结果，将执行结果截图。

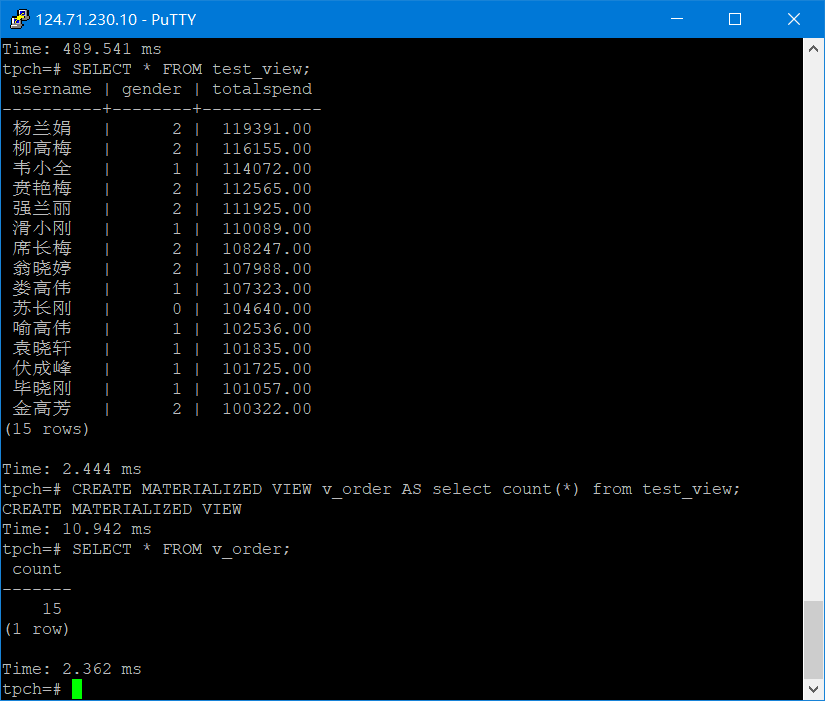
SELECT \* FROM vi\_order;

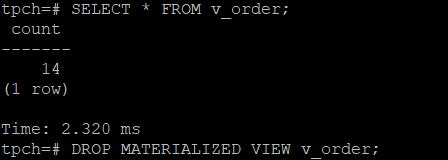
5. 对表进行操作后，刷新增量物化视图，查询物化视图结果，将执行结果截图。

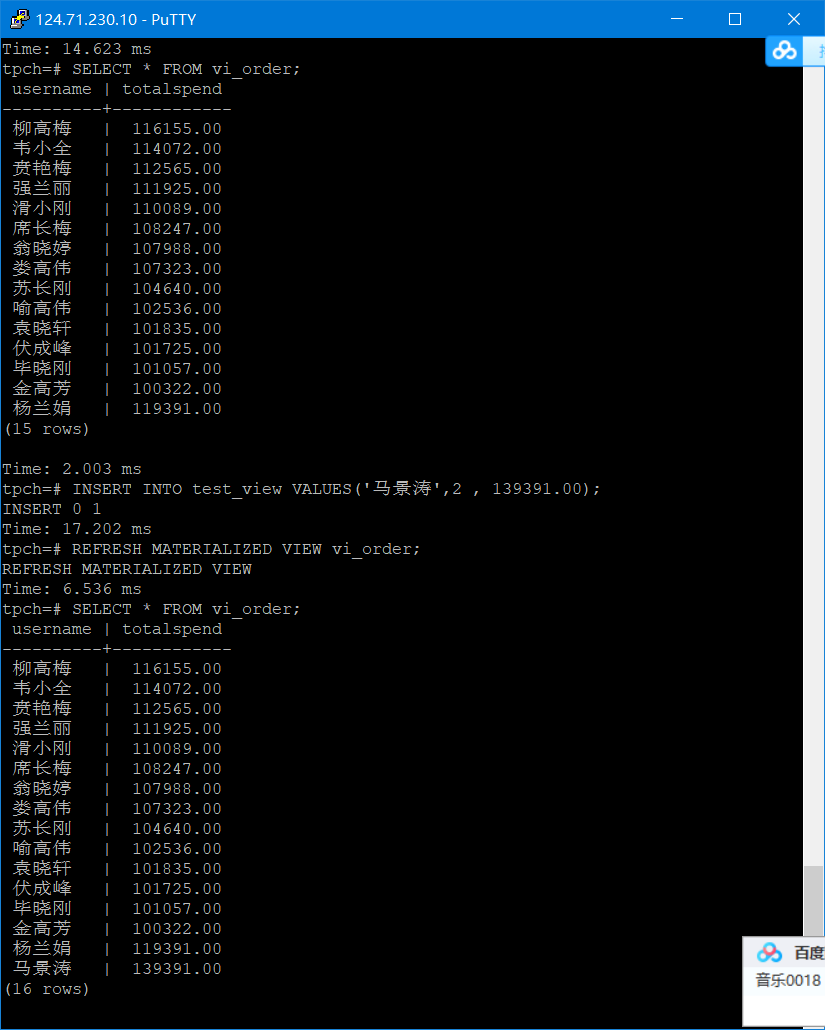
SELECT \* FROM vi\_order;











实践思考题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？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？

基于行的存储是将数据组织成多个行，在处理相同实体的两个或多个列的查询时能够取得更快的速度，而且可以提高更新、插入和删除操作的速度。

基于列的访问存在的缺点是载入速度通常比较慢，因为源数据在外部来源中是以行或者记录的形式表示的。这样做的优点是针对某个列中的值进行简单查询的速度非常快，需要的内部存储资源最少。这种体系结构在处理数据仓库使用的海量数据时没有问题，但不适合需要进行大量以行的方式进行访问和更新操作的联机事物处理。

实践思考题2：全量物化视图与增量物化视图有哪些差别？

物化视图是包括一个查询结果的数据库对象，它是远程数据的的本地副本，或者用来生成基于数据表求和的汇总表。物化视图存储基于远程表的数据，也可以称为快照。