openGauss AI特性创新实践课



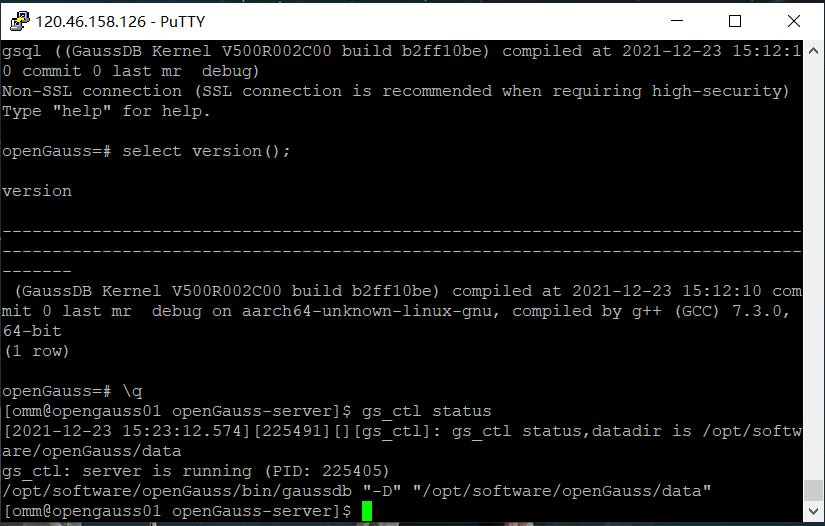
华为技术有限公司

# 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：

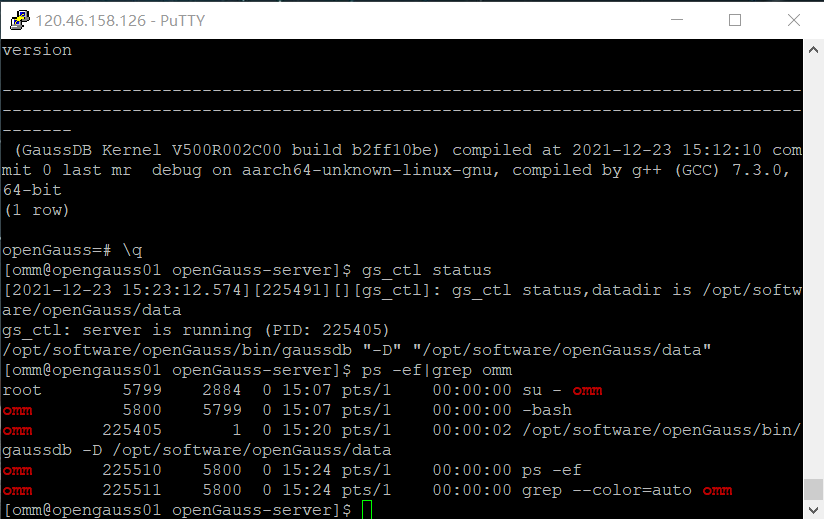
任务一：数据库状态验证

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）



实验思考题：为什么需要通过源码编译，安装数据库？

答：原因如下

1.通过源码编译能够更加了解数据库的安装过程，底层的实现原理等，是一个非常不错的学习过程。

2.源码编译安装数据库的方法是需要自己手动配置一些参数，这样子做的好处是可以更加灵活地处理数据库的安装问题，具有更高的灵活性。

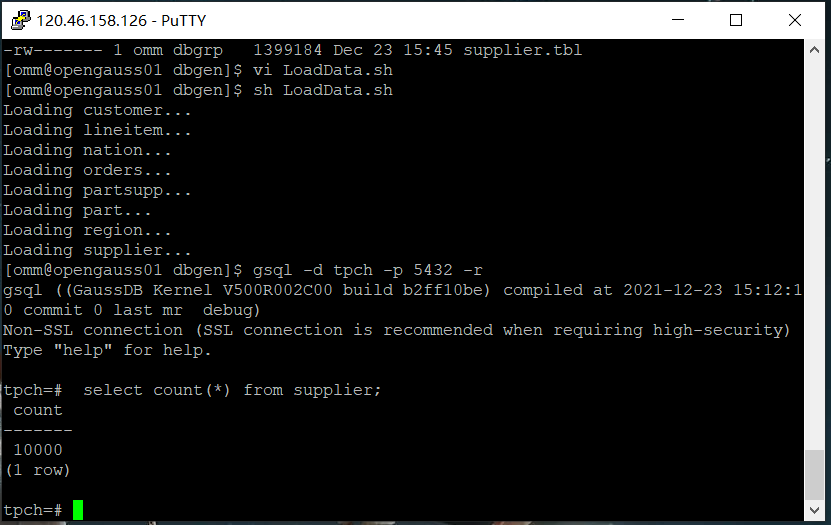
3.源码编译安装数据库的方法还可以解决更换硬盘和系统时文件转移的问题，能够更加快速的拷贝文件内容。

# 关卡二、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：

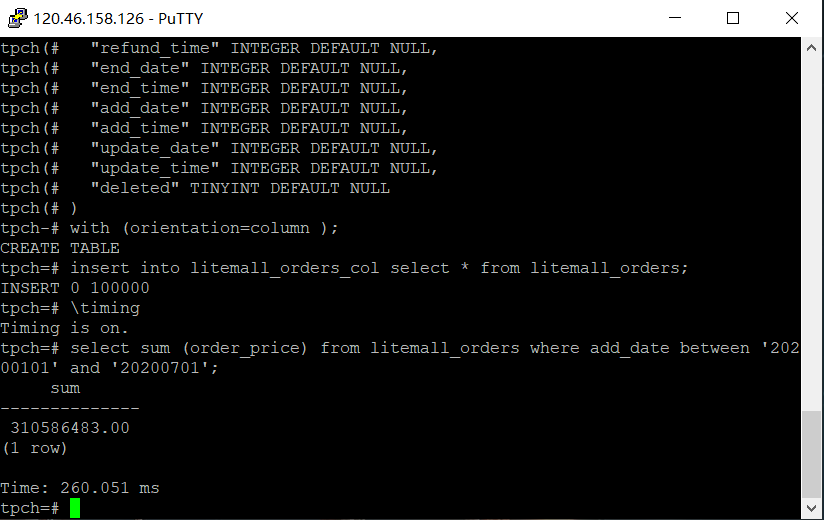
select count(\*) from supplier;;



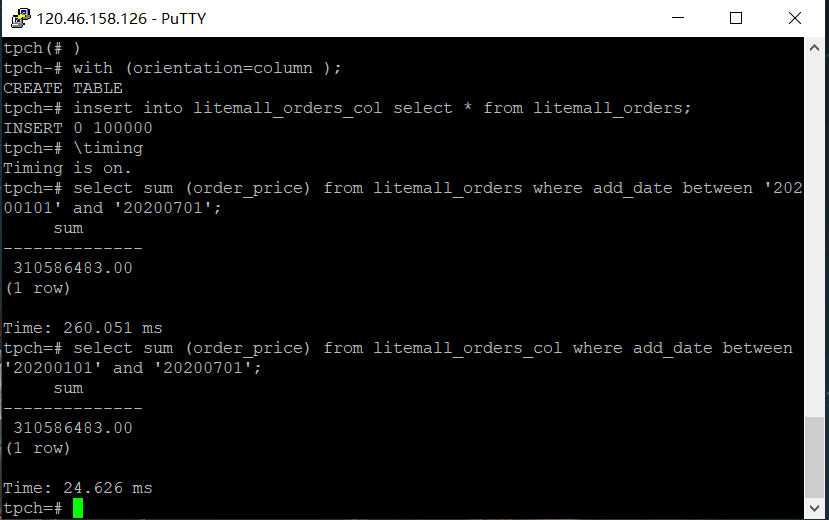
任务二：行存表与列存表执行效率对比

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';

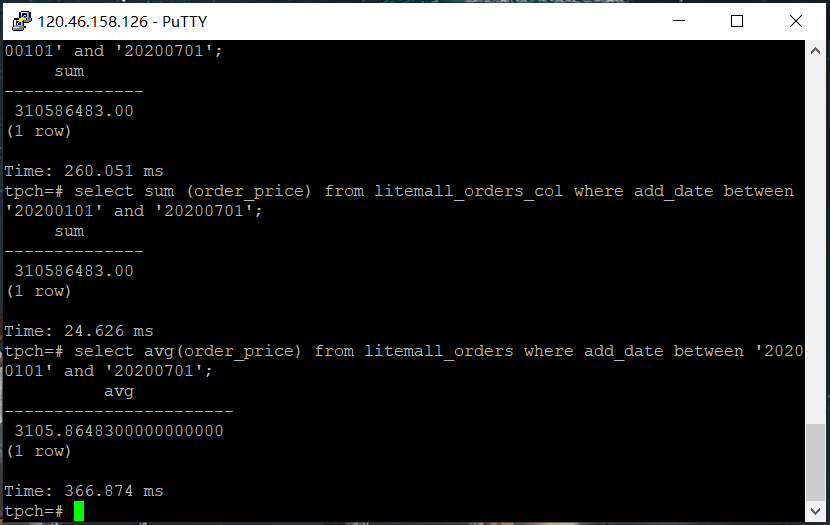


select sum (order\_price) from litemall\_orders\_col where add\_date between '20200101' and '20200701';



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';

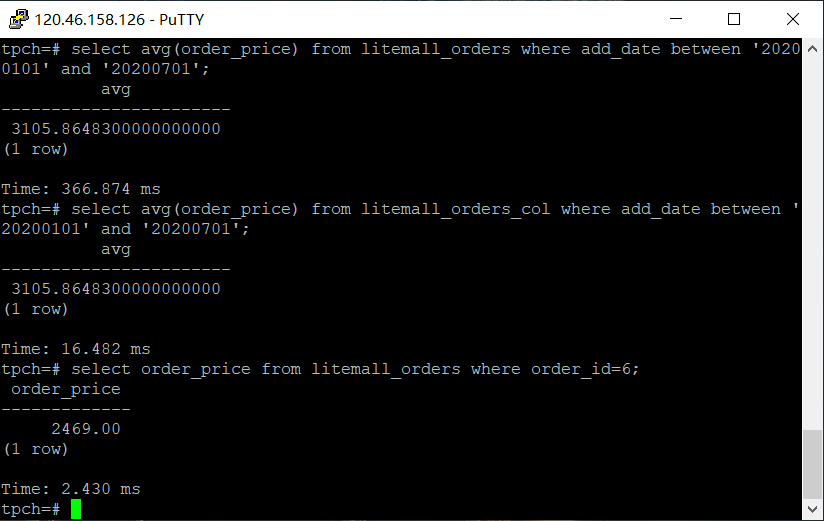


select avg (order\_price) from litemall\_orders\_col where add\_date between '20200101' and '20200701';

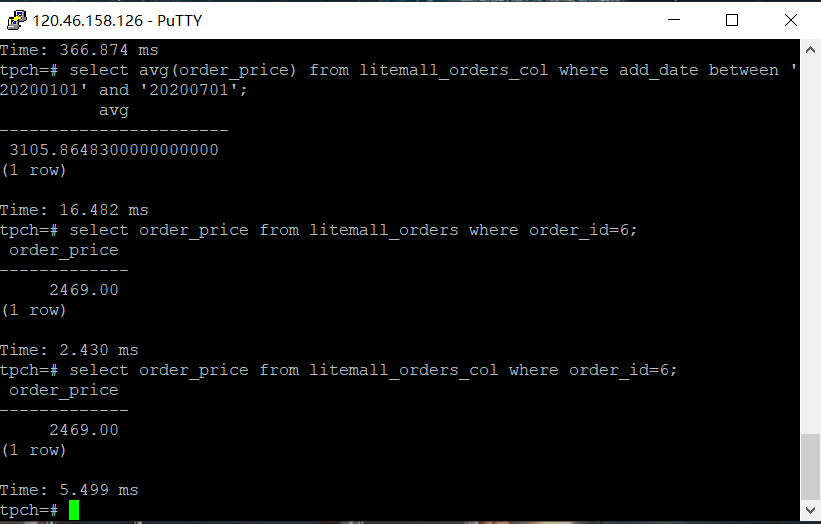


3. 查询litemall\_orders行存表与litemall\_orders\_col列存表中order\_id为6的order\_price的值，并对比执行效率截图。

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

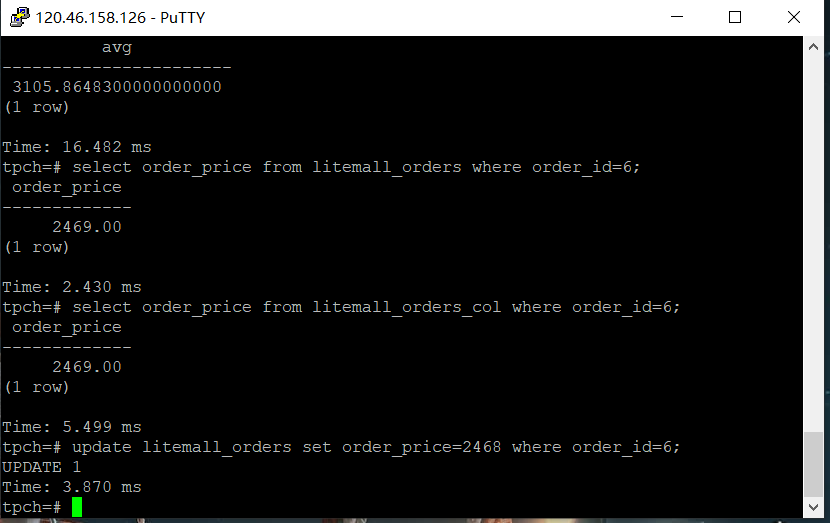


select order\_price from litemall\_orders\_col where order\_id=6;

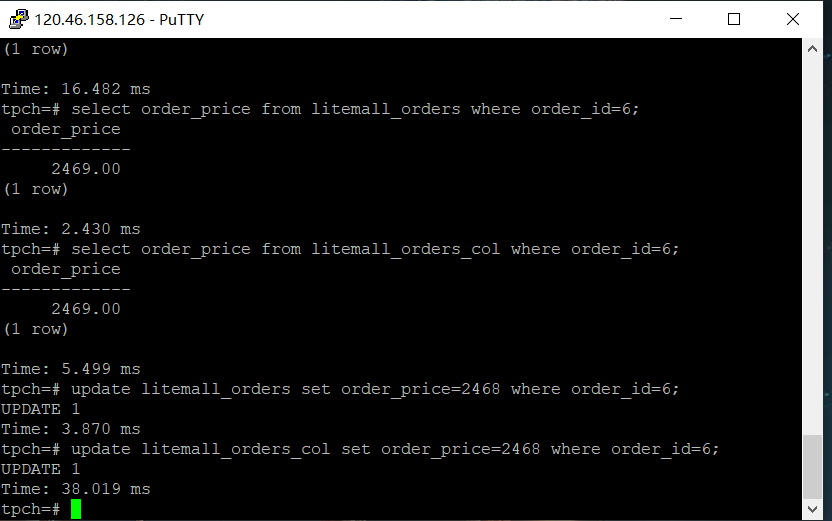


4. 将litemall\_orders行存表与litemall\_orders\_col列存表中order\_id为6的order\_price修改为2468，并对比执行效率截图。

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



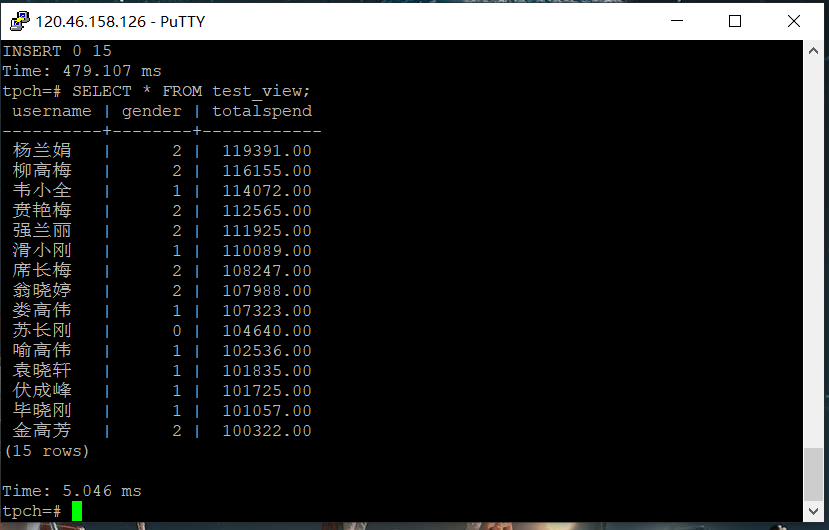
update litemall\_orders\_col set order\_price=2468 where order\_id=6;



任务三：物化视图的使用

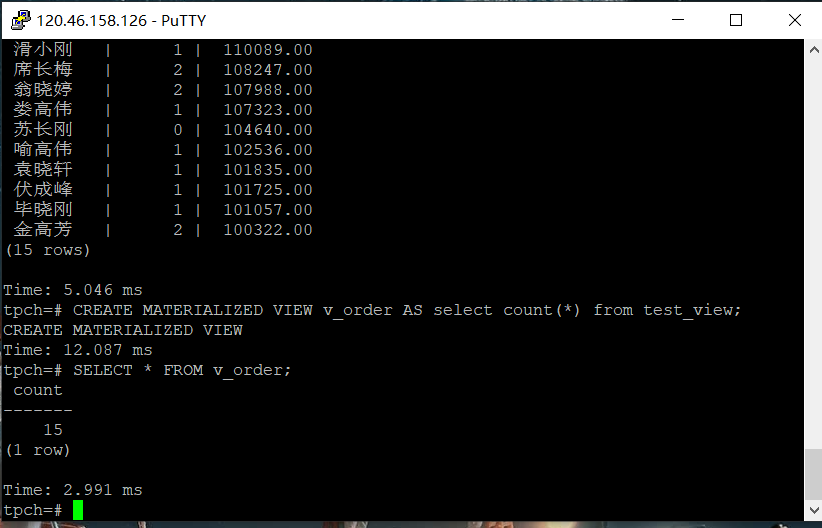
1. 创建物化视图所需要的表后，对表内容进行查询，对查询结果截图：

SELECT \* FROM test\_view;



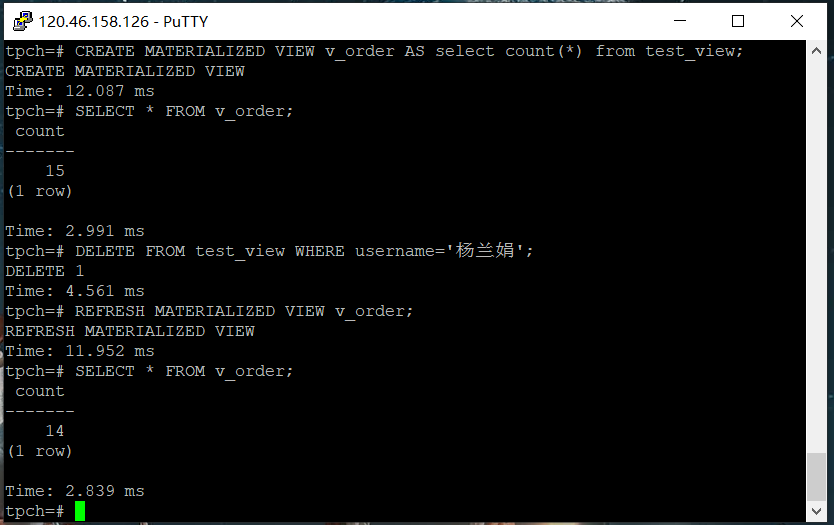
2. 使用物化视图统计人数，查询物化视图结果，将执行结果截图。

SELECT \* FROM v\_order;



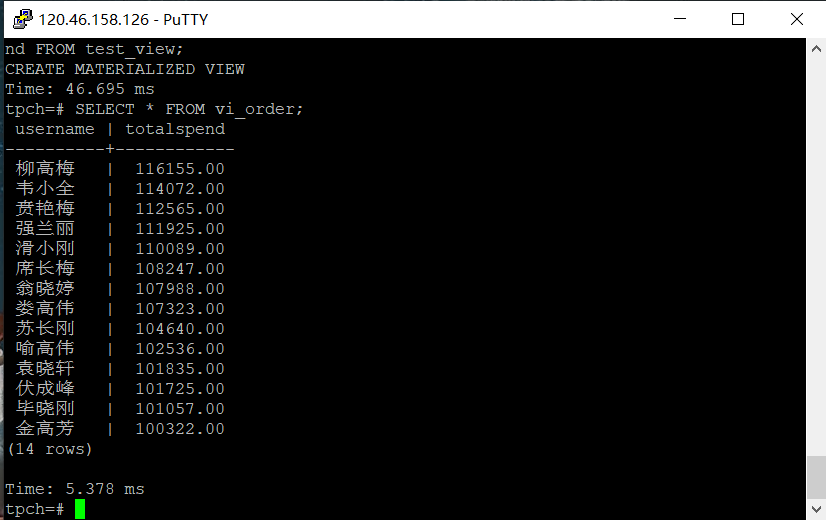
3. 对表进行操作后，刷新物化视图，查询物化视图结果，将执行结果截图。

SELECT \* FROM v\_order;



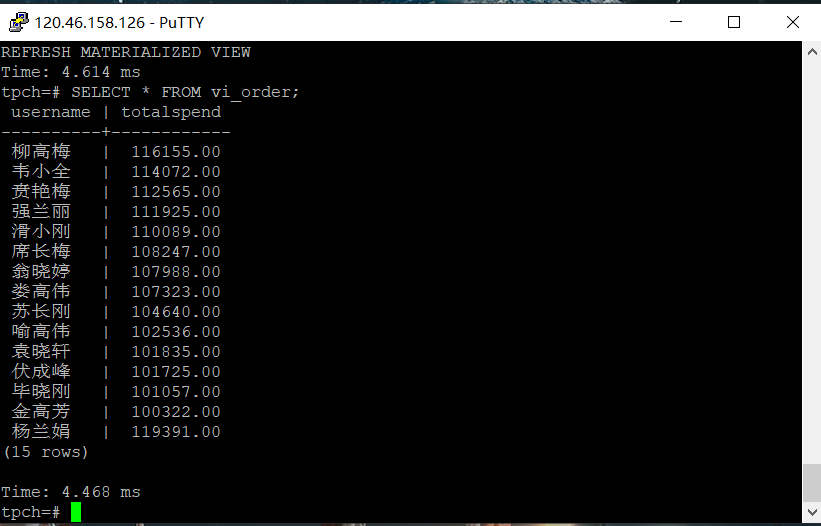
4. 创建增量物化视图，查询物化视图结果，将执行结果截图。

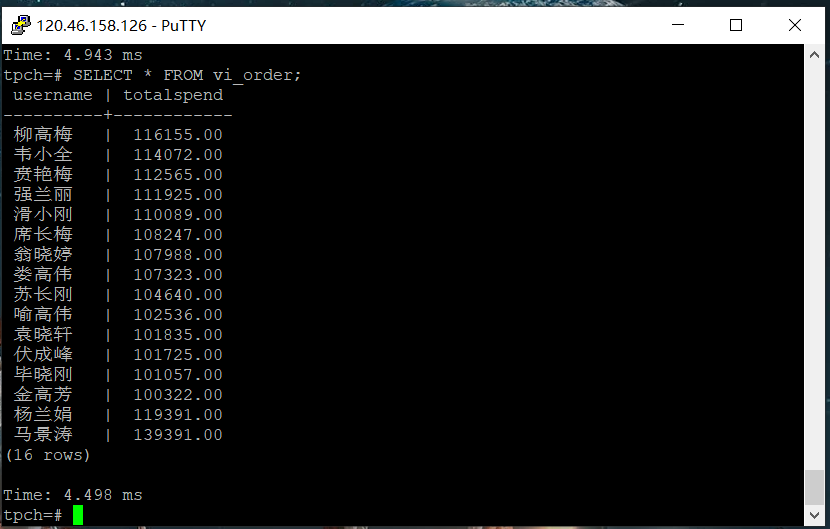
SELECT \* FROM vi\_order;



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？

答：（1）原因是行存表与列存表在存储结构上的不同。一般来说，使用select语句时，即使只是涉及到了几列，但是所有的数据都会被读取。

（2）当表的字段比较多的时候，同时查询中涉及的列不是很多的情况下就可以使用列存表。如果表的字段比较少的时候，同时需要查询大部分字段时，就可以使用行存表。

实践思考题2：全量物化视图与增量物化视图有哪些差别？

答：全量物化视图只会对创建好的物化视图进行全量更新，不支持增量更新。同时不可以对全量物化视图进行指定的创建。

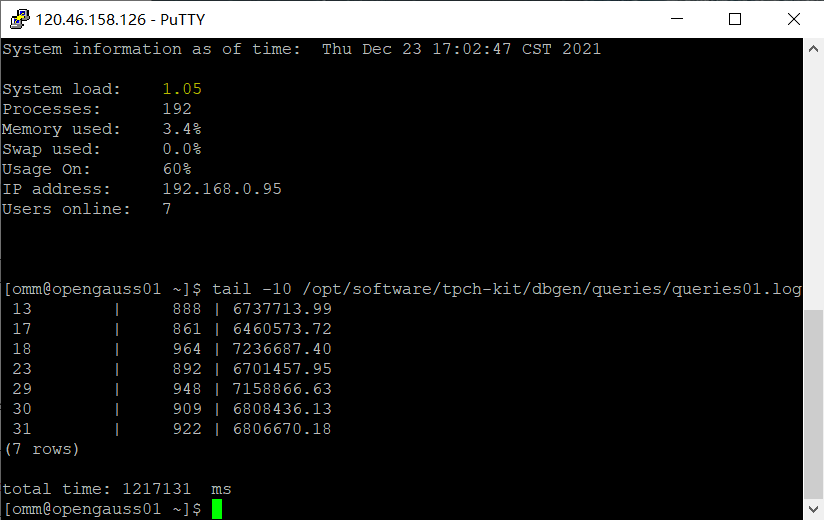
而增量物化视图是对物化视图进行增量刷新，同时需要用户手动执行才可以进行物化视图的更新。目前增量物化视图的使用场景比较少，仅仅支持基表扫描语句或者UNION ALL语句。

# 关卡三、openGauss的AI4DB特性应用

任务一：使用X-Tuner进行参数优化

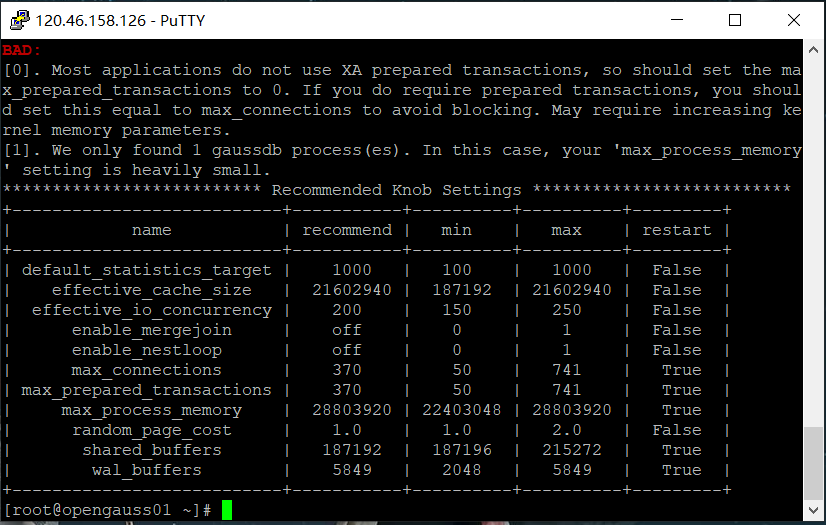
1. 执行TPCH脚本，获得测试时间，将执行结果截图：

gsql -d tpch -p 5432 -r -f /opt/software/tpch-kit/dbgen/queries/queries.sql > /opt/software/tpch-kit/dbgen/queries/queries01.log



2. 使用root用户，执行X-Tuner进行参数建议优化，将执行结果截图

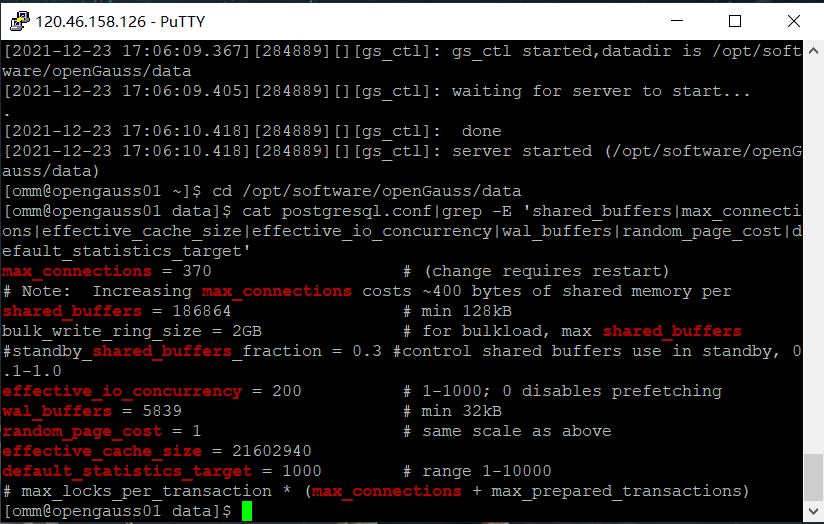
gs\_xtuner recommend --db-name tpch --db-user omm --port 5432 --host 127.0.0.1 --host-user omm



3.重启完成后，获取参数值：

cd /opt/software/openGauss/data

cat postgresql.conf|grep -E 'shared\_buffers|max\_connections|effective\_cache\_size|effective\_io\_concurrency|wal\_buffers|random\_page\_cost|default\_statistics\_target'



任务二：使用Index-advisor对select 查询语句进行优化，并通过对比执行计划，得到优化前后的不同。

1. 使用explain，对查询2020年3月订单表收入并进行排序的SQL加以分析，将结果截图。

EXPLAIN

SELECT ad.province AS province, SUM(o.actual\_price) AS GMV

FROM litemall\_orders o,

address\_dimension ad,

date\_dimension dd

WHERE o.address\_key = ad.address\_key

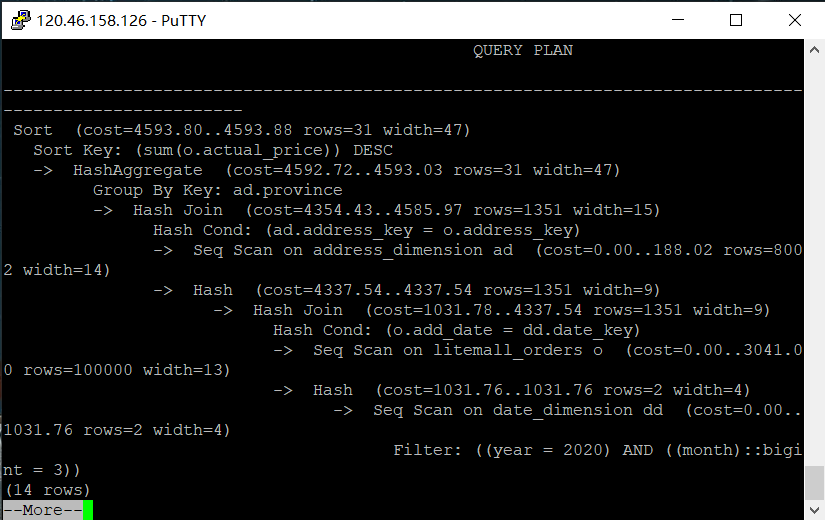
AND o.add\_date = dd.date\_key

AND dd.year = 2020

AND dd.month = 3

GROUP BY ad.province

ORDER BY SUM(o.actual\_price) DESC;



2. 使用索引推荐功能，对查询语句进行推荐，将执行结果截图。

select \* from gs\_index\_advise('

SELECT ad.province AS province, SUM(o.actual\_price) AS GMV

FROM litemall\_orders o,

address\_dimension ad,

date\_dimension dd

WHERE o.address\_key = ad.address\_key

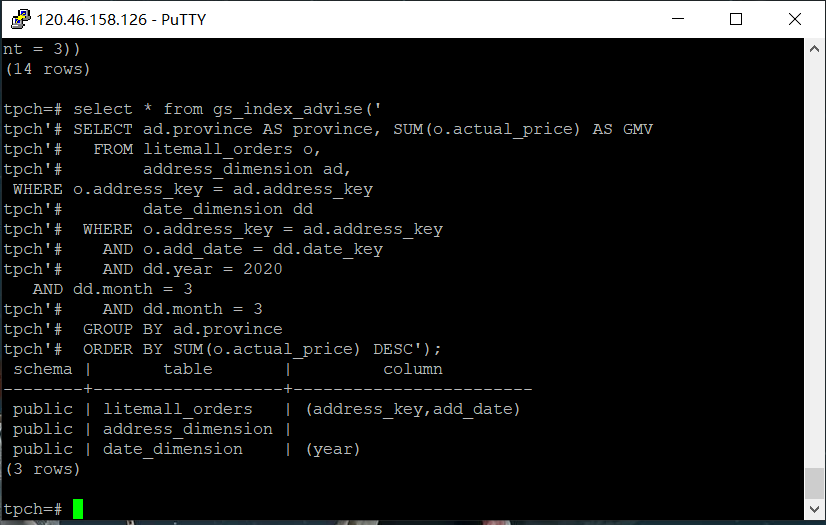
AND o.add\_date = dd.date\_key

AND dd.year = 2020

AND dd.month = 3

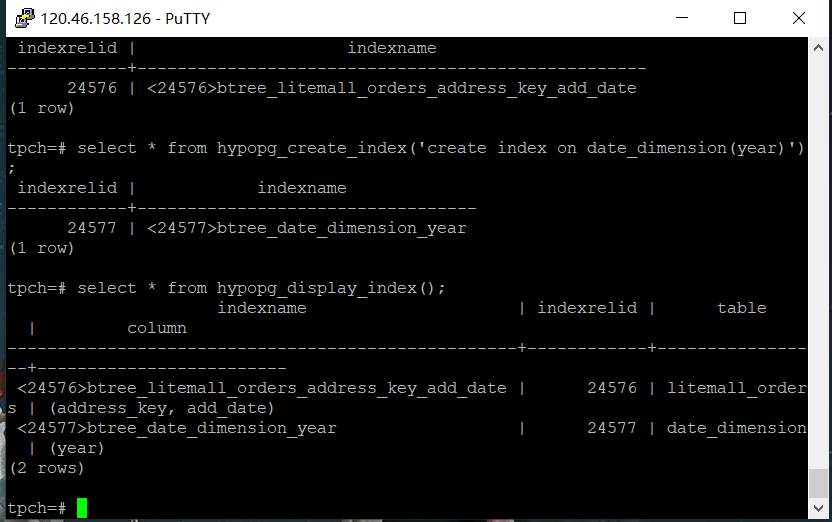
GROUP BY ad.province

ORDER BY SUM(o.actual\_price) DESC');



3. 查看创建的虚拟索引列，将执行结果截图。

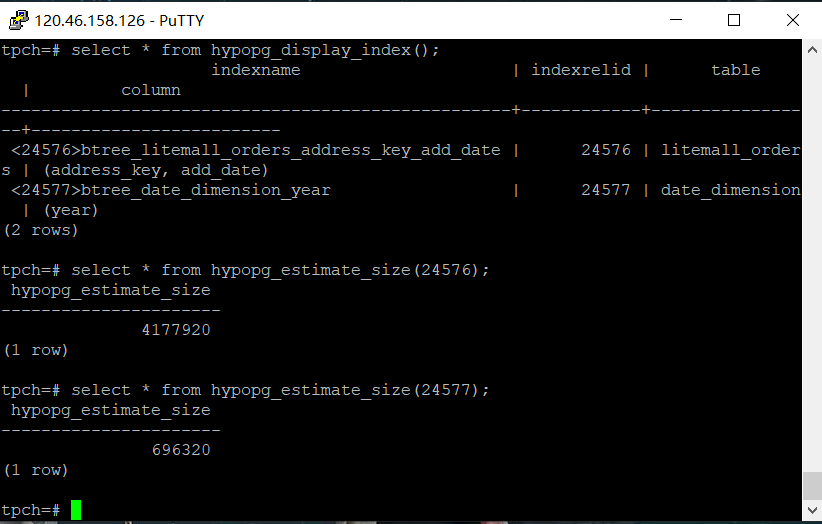
select \* from hypopg\_display\_index();



4. 获取索引虚拟列大小结果（单位为：字节），将执行结果截图。

select \* from hypopg\_estimate\_size(16715);

select \* from hypopg\_estimate\_size(16716);



5.再次使用explain，对该SQL加以分析，将执行结果截图。

EXPLAIN

SELECT ad.province AS province, SUM(o.actual\_price) AS GMV

FROM litemall\_orders o,

address\_dimension ad,

date\_dimension dd

WHERE o.address\_key = ad.address\_key

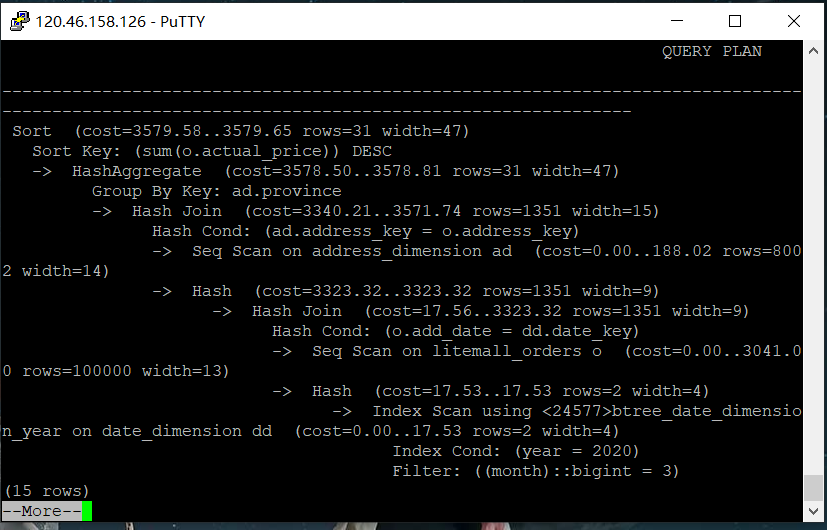
AND o.add\_date = dd.date\_key

AND dd.year = 2020

AND dd.month = 3

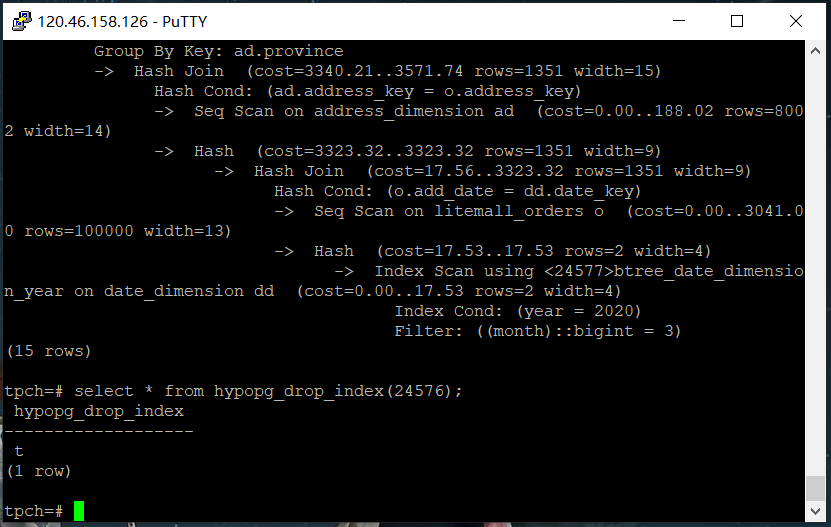
GROUP BY ad.province

ORDER BY SUM(o.actual\_price) DESC;



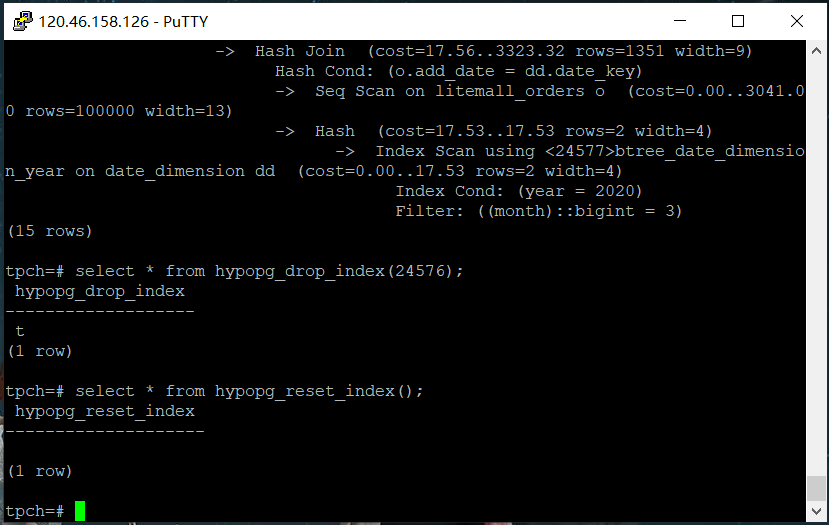
6. 删除某一个索引虚拟列，将执行结果截图。

select \* from hypopg\_drop\_index(16715);



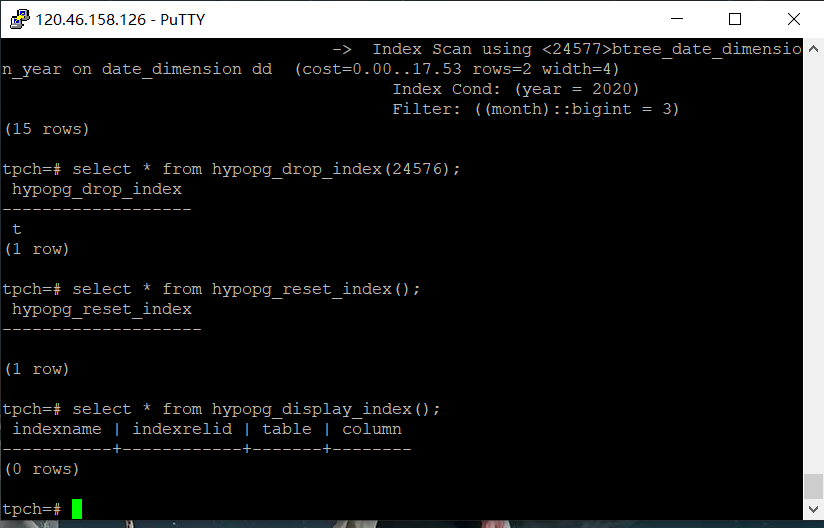
7. 删除某一个索引虚拟列，将执行结果截图。

select \* from hypopg\_reset\_index();



8. 查看索引虚拟列，将执行结果截图。

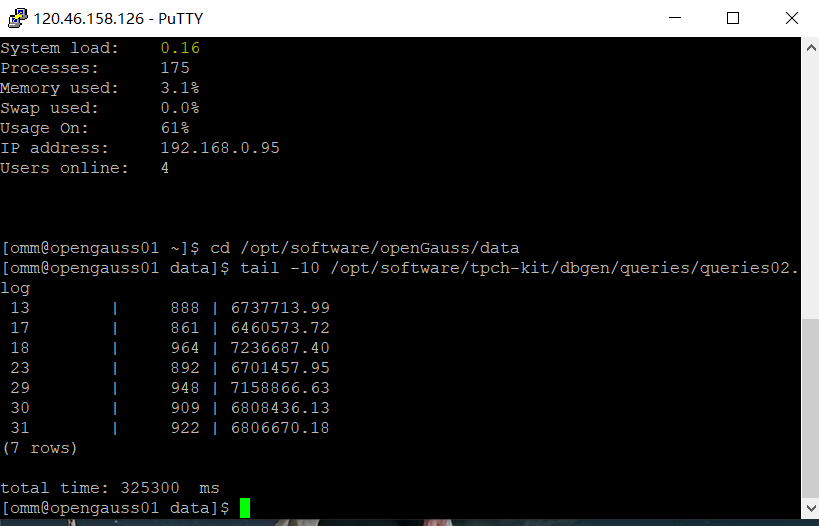
select \* from hypopg\_display\_index();



任务三：通过创建索引，对queries.sql中的SQL语句进行优化，并对比优化前后queries.sql执行的时间。

1. 重新执行queries.sql查询，将执行结果截图：

gsql -d tpch -p 5432 -r -f /opt/software/tpch-kit/dbgen/queries/queries.sql > /opt/software/tpch-kit/dbgen/queries/queries02.log



挑战一：进一步优化queries.sql中的查询语句，使得前后执行时间出现倍数级的提升。

1. 重新执行queries.sql查询，将执行结果截图：

gsql -d tpch -p 5432 -r -f /opt/software/tpch-kit/dbgen/queries/queries.sql > /opt/software/tpch-kit/dbgen/queries/queries03.log



实践思考题1：根据X-Tuner给出的参数优化，在哪些参数上进行了优化，为何要对这些参数进行优化？

答：优化参数如下

1.temp\_file\_size：产生的临时文件数量，使用过多的临时文件会导致性能不佳，如果可能的话，需要提高work\_mem参数的配置。

2.workload\_type：根据数据库统计信息，推测当前负载类型，分为AP、TP以及HTAP三种类型。

3.checkpoint\_avg\_sync\_time：数据库在checkpoint 时，平均每次同步刷新数据到磁盘的时长。

4.cache\_hit\_rate：shared\_buffer 的缓存命中率，表明当前workload使用缓存的效率。

5.read\_write\_ratio：数据库作业的读写比例。

6.search\_modify\_ratio：数据库作业的查询与修改数据的比例。

7.load\_average：平均每个CPU核心在1分钟、5分钟以及15分钟内的负载。8.ap\_index：表明当前workload的AP指数，取值范围是0到10，该数值越大，表明越偏向于数据分析与检索。

实践思考题2：索引的使用，对于执行SQL有什么好处？除了使用索引和参数外，还有哪些方面可以对数据库进行优化？

答：1.使用索引可以保证数据库中数据的唯一性，也就是每一行数据的唯一性。

2.使用索引可以加快数据的检索速度，加快数据的读取。

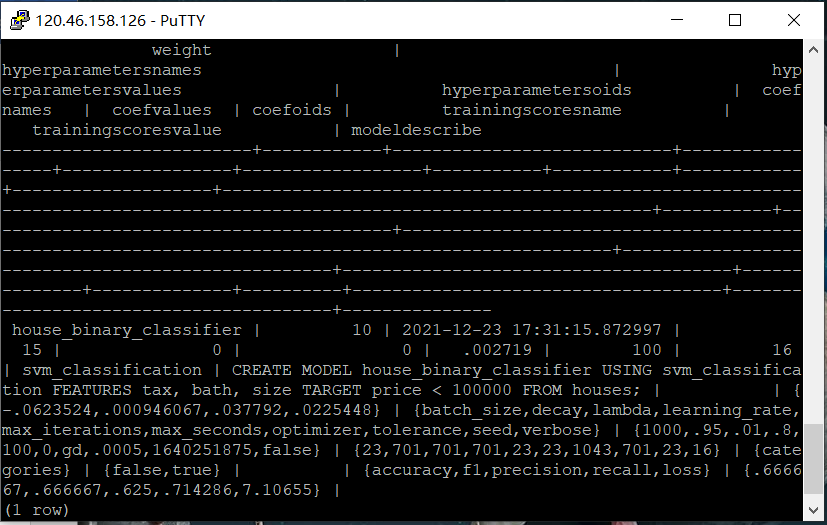
3.减少查询与分组排序的时间。

4.在查询的过程中，使用优化隐藏器，提高系统的性能。

# 关卡四、openGauss的DB4AI特性应用

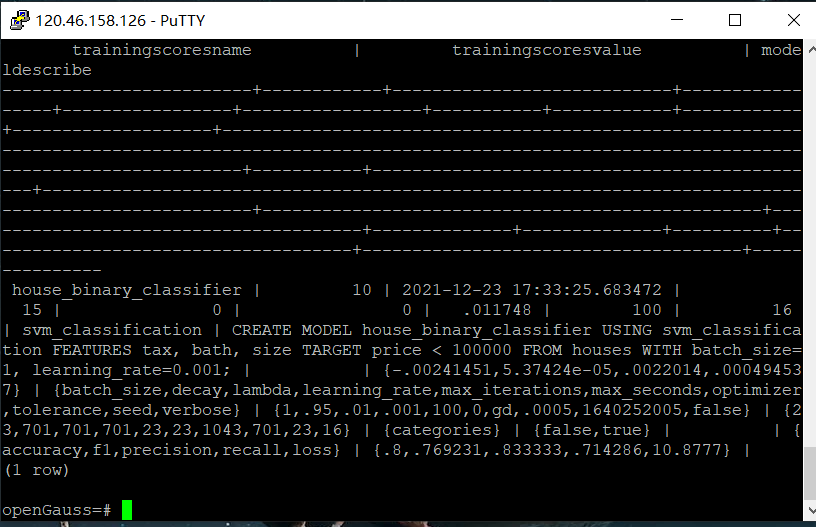
任务一：在gs\_model\_warehouse系统表中查看训练后的模型信息，将执行结果截图：

postgres=# SELECT \* FROM gs\_model\_warehouse WHERE modelname = 'house\_binary\_classifier';



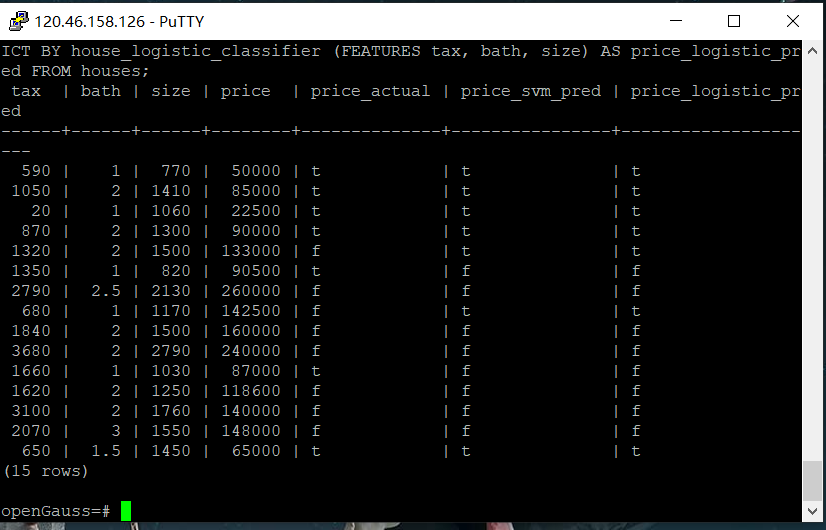
任务二：观察新模型的信息，将执行结果截图。

postgres=# SELECT \* FROM gs\_model\_warehouse WHERE modelname = 'house\_binary\_classifier';



任务三：利用训练好的逻辑回归模型预测数据，并与SVM算法进行比较，将执行结果截图。

postgres=# SELECT tax, bath, size, price, price < 100000 AS price\_actual, PREDICT BY house\_binary\_classifier (FEATURES tax, bath, size) AS price\_svm\_pred, PREDICT BY house\_logistic\_classifier (FEATURES tax, bath, size) AS price\_logistic\_pred FROM houses;



实践思考题1：分类模型与回归模型有何不同？

答：定量输出称为回归，或者说是连续变量预测；定性输出称为分类，或者说是离散变量预测。即二者的不同在于输出变量的类型不同。

实践思考题2：什么是SVM算法？

答：SVM是一个二元分类算法，线性分类和非线性分类都支持。 经过演进，现在也可以支持多元分类，同时经过扩展，也能应用于回归问题。

实践思考题3：分类问题有哪些评价指标，请分别说明他们的含义？

答：1.准确率，正确的结果占总样本的百分比

2.精准率，即查准率，它是针对预测结果而言的，它的含义是在所有被预测为正的样本中实际为正的样本的概率。

3. 召回率，即查全率，它是针对原样本而言的，它的含义是在实际为正的样本中被预测为正样本的概率。

4. P-R曲线（Precision Recall Curve）正是描述精确率/召回率变化的曲线。

5. ROC曲线，又称接受者操作特征曲，其用于评价模型的预测能力。两个指标就是真正率TPR和假正率FPR。

6.AUC，即曲线下面积，是处于ROC Curve下方的那部分面积的大小。

实践思考题4：回归问题有哪些评价指标，请分别说明他们的含义？

答：1. 平均绝对误差，平均绝对误差就是指预测值与真实值之间平均相差多大。

2. 均方误差，观测值与真值偏差的平方和与观测次数的比值

3. 分母理解为原始数据的离散程度，分子为预测数据和原始数据的误差，二者相除可以消除原始数据离散程度的影响其实“决定系数”是通过数据的变化来表征一个拟合的好坏。