openGauss AI特性创新实践课



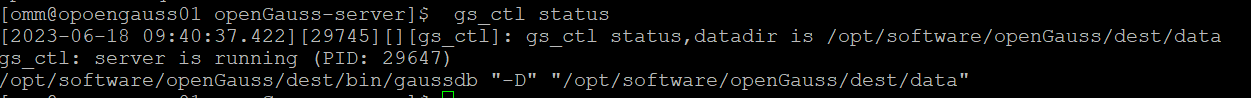
华为技术有限公司

# 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：

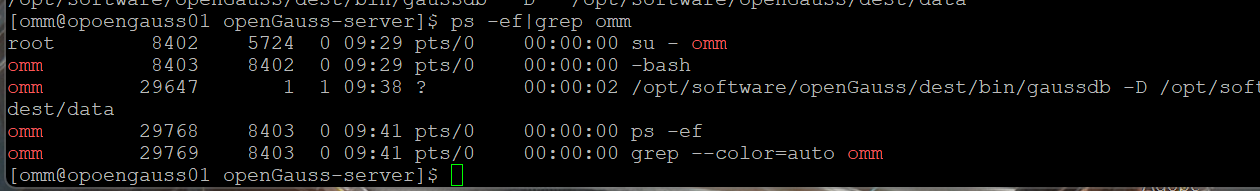
任务一：数据库状态验证

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？

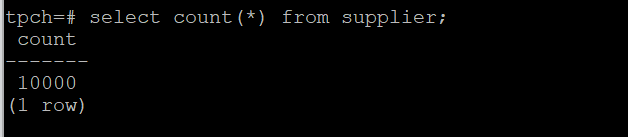
通过源码编译安装数据库，可以提供更大的灵活性和定制化配置选项。源代码通常包含各种编译选项，可以根据具体需求启用或禁用某些功能，调整性能参数，或者进行特定平台的优化。这使得数据库能够满足特定应用场景的需求，提供最佳的性能和功能组合。

# 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：

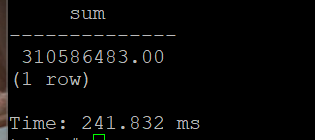
select count(\*) from supplier;;



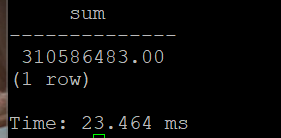
任务二：行存表与列存表执行效率对比

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';

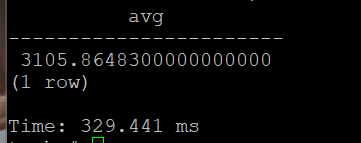


select sum (order\_price) from litemall\_orders\_col where add\_date between '20200101' and '20200701';

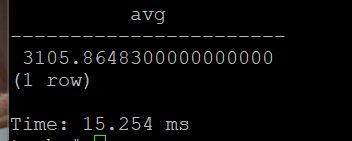


2. 2020年上半年litemall\_orders行存表与litemall\_orders\_col列存表中的order\_price的平均值查询，并对比执行效率截图

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

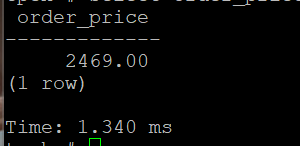


select avg (order\_price) from litemall\_orders\_col where add\_date between '20200101' and '20200701';

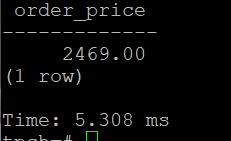


3. 查询litemall\_orders行存表与litemall\_orders\_col列存表中order\_id为6的order\_price的值，并对比执行效率截图。

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



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

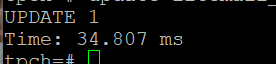


4. 将litemall\_orders行存表与litemall\_orders\_col列存表中order\_id为6的order\_price修改为2468，并对比执行效率截图。

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

屏幕截图 2023-06-18 101912

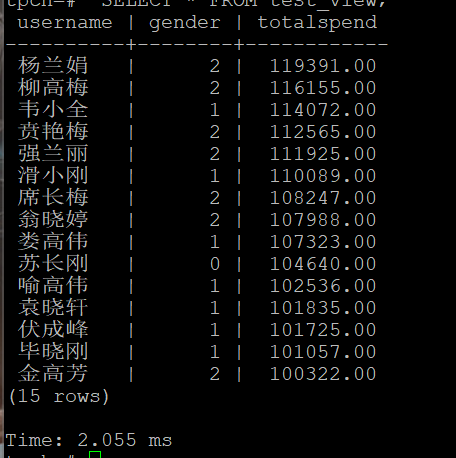
update litemall\_orders\_col set order\_price=2468 where order\_id=6;



任务三：物化视图的使用

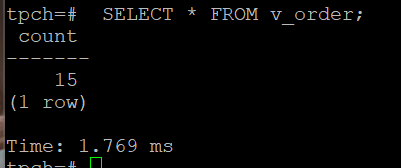
1. 创建物化视图所需要的表后，对表内容进行查询，对查询结果截图：

SELECT \* FROM test\_view;



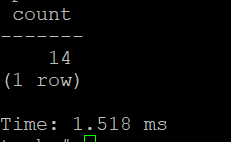
2. 使用物化视图统计人数，查询物化视图结果，将执行结果截图。

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。

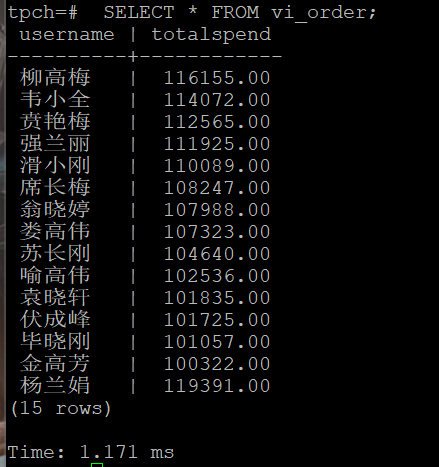
SELECT \* FROM v\_order;



1. 创建增量物化视图，查询物化视图结果，将执行结果截图。

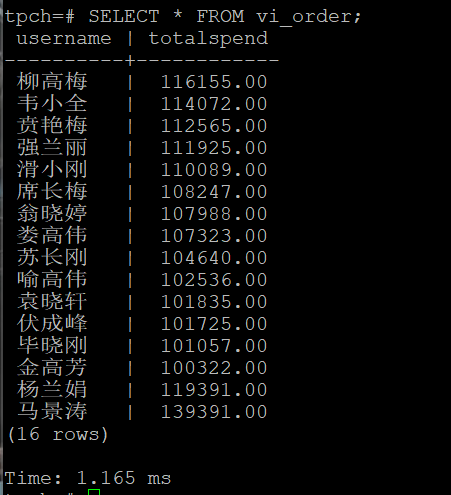


SELECT \* FROM vi\_order;



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？

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主要是由于它们的数据存储方式和查询处理方式不同。

行存表是将数据按行存储在磁盘上，每一行数据都是一个完整的记录，包含了表的所有列。当执行SQL查询时，行存表需要读取整行数据，包括那些并不在查询结果中的列。这种存储方式在执行一些需要读取整行数据的操作时比较高效，比如对某个特定记录进行更新或者删除操作。

列存表则是将数据按列存储在磁盘上，每个列都存储在独立的列存储区域中。当执行SQL查询时，列存表只需要读取所需的列数据，而不需要读取整行数据。这种存储方式在执行一些需要聚合计算、统计和分析的操作时比较高效，比如对某个列进行求和、计数或者排序操作。

实践思考题2：全量物化视图与增量物化视图有哪些差别？

1、全量物化视图存储整个查询结果，增量物化视图存储查询结果的摘要信息。

2、全量物化视图更新时需要重新计算整个查询结果，增量物化视图只需要计算变化的部分数据。

3、全量物化视图的更新成本和时间消耗较高，增量物化视图的更新成本和时间消耗较低。

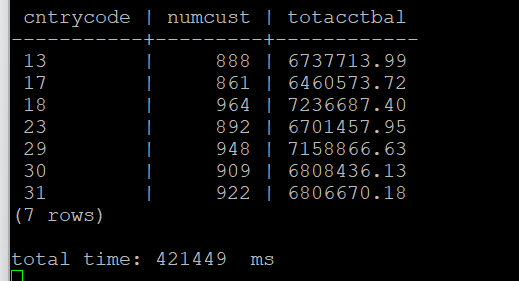
4、全量物化视图适用于静态或者变化较少的数据集，增量物化视图适用于数据变化频繁的场景。

# 关卡三、openGauss的AI4DB特性应用

任务一：使用X-Tuner进行参数优化

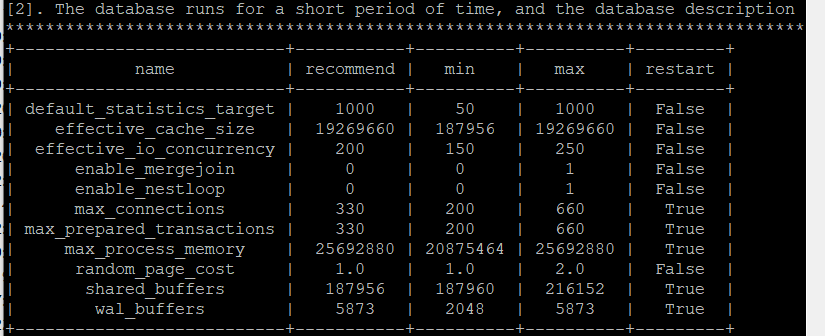
1. 执行TPCH脚本，获得测试时间，将执行结果截图：

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

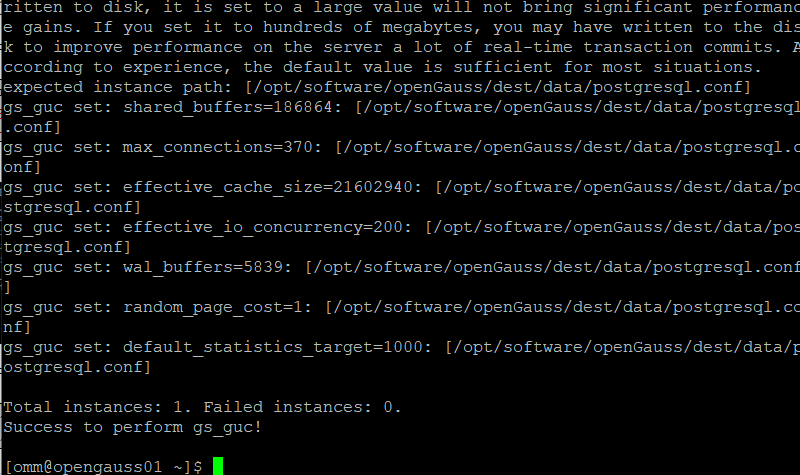
gs\_xtuner recommend --db-name tpch --db-user omm --port 5432 --host 127.0.0.1 --host-user omm



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

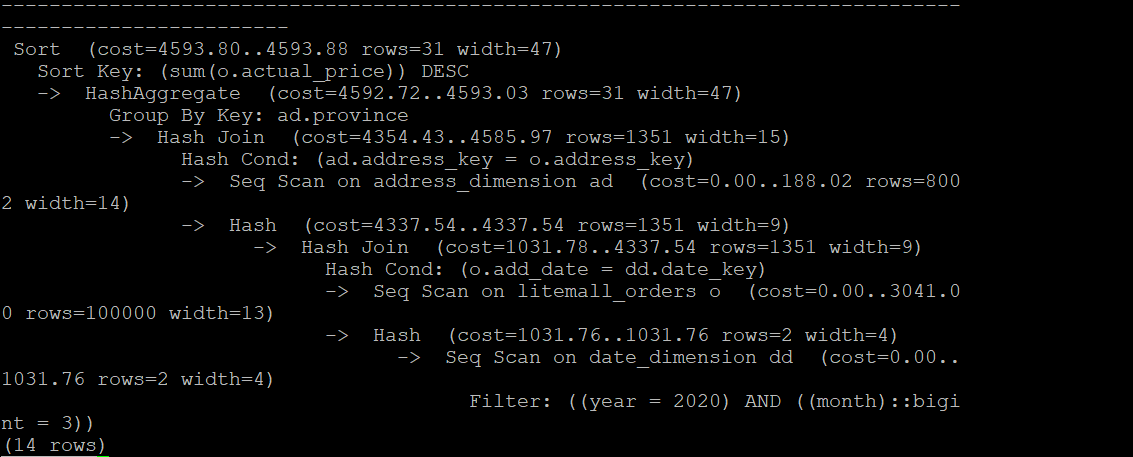
AND o.add\_date = dd.date\_key

AND dd.year = 2020

AND dd.month = 3

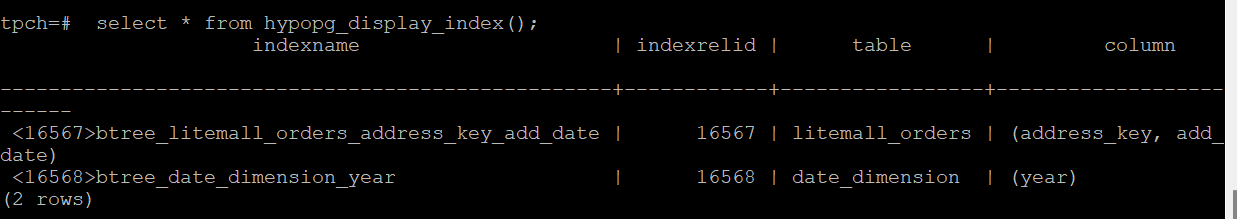
GROUP BY ad.province

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。

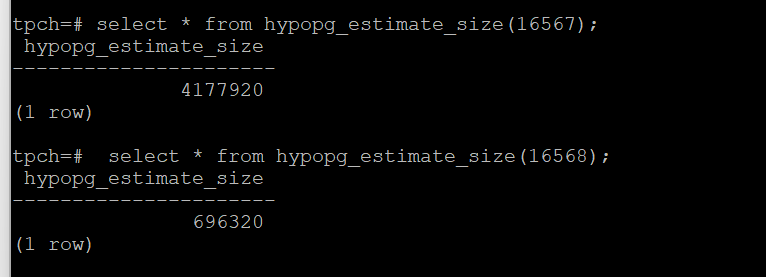
select \* from hypopg\_display\_index();



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

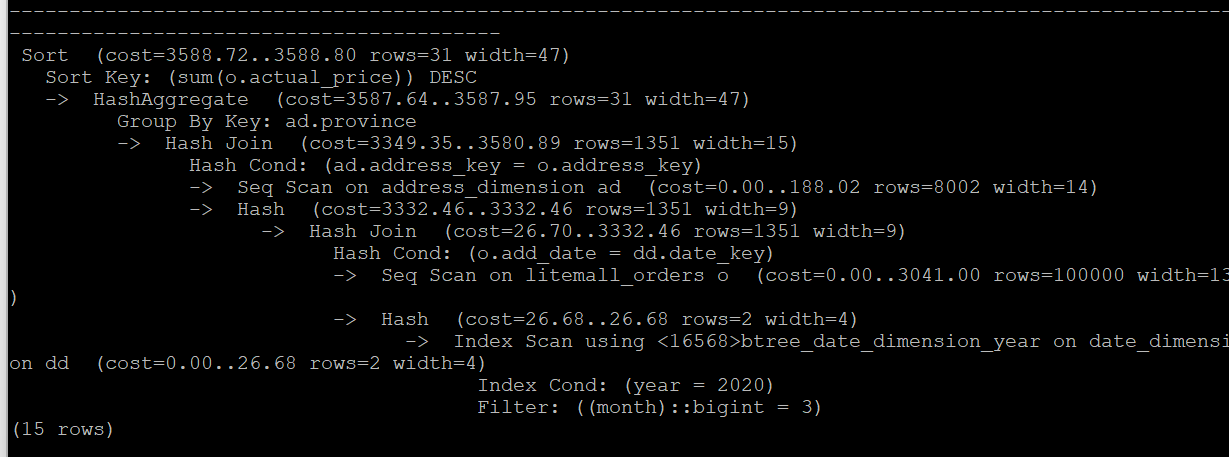
AND o.add\_date = dd.date\_key

AND dd.year = 2020

AND dd.month = 3

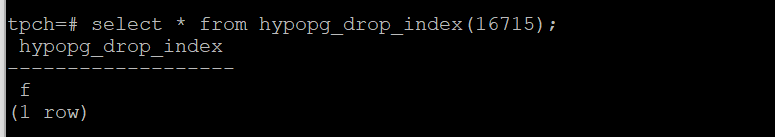
GROUP BY ad.province

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



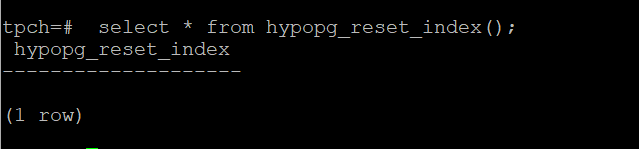
6. 删除某一个索引虚拟列，将执行结果截图。

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



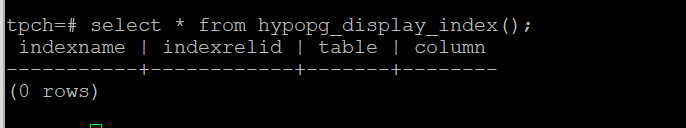
7. 删除某一个索引虚拟列，将执行结果截图。

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



8. 查看索引虚拟列，将执行结果截图。

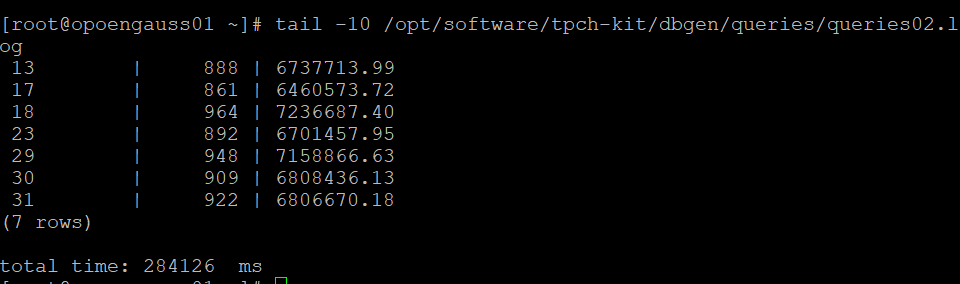
select \* from hypopg\_display\_index();



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：

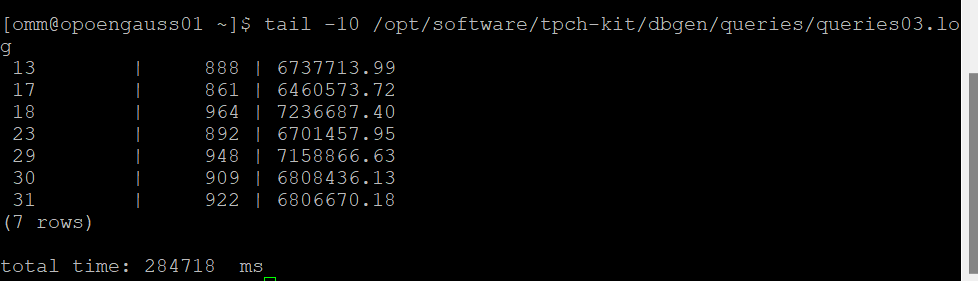
gsql -d tpch -p 5432 -r -f /opt/software/tpch-kit/dbgen/queries/queries.sql > /opt/software/tpch-kit/dbgen/queries/queries02.log



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？

X-Tuner 是一款数据库集成的参数调优工具，通过结合深度强化学习和全局搜索算法等AI技术，实现在无需人工干预的情况下，获取最佳数据库参数配置。它包含三种运行模式：recommend、train 和 tune。

在 recommend 模式下，X-Tuner 通过用户指定的用户名等信息登录到数据库环境中，获取当前正在运行的 workload 特征信息，根据上述特征信息生成参数推荐报告。报告当前数据库中不合理的参数配置和潜在风险等；输出根据当前正在运行的 workload 行为和特征；输出推荐的参数配置。

在 train 模式下，X-Tuner 通过用户提供的 benchmark 信息，不断地进行参数修改和 benchmark 的执行。通过反复的迭代过程，训练强化学习模型，以便用户在后面通过 tune 模式加载该模型进行调优。

在 tune 模式下，X-Tuner 使用优化算法进行数据库参数的调优。当前支持两大类算法，一种是深度强化学习，另一种是全局搜索算法（全局优化算法）。深度强化学习模式要求先运行 train 模式，生成训练后的调优模型，而使用全局搜索算法则不需要提前进行训练，可以直接进行搜索调优。

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？

索引是一种数据结构，它能帮助我们快速匹配查找到需要的数据行，是数据库性能优化最常用的工具之一。它的作用相当于超市里的导购员、书本里的目录。

使用索引有很多好处，例如：

通过唯一性索引（unique）可确保数据的唯一性；

加快数据的检索速度；

加快表之间的连接；

减少分组和排序时间；

使用优化隐藏器提高系统性能。

除了使用索引和参数外，还有其他方面可以对数据库进行优化。例如，可以通过维护索引来提高查询性能并减少资源消耗。可以通过重新组织和重新生成索引来减少索引碎片并增加页面密度。

# 关卡四、openGauss的DB4AI特性应用

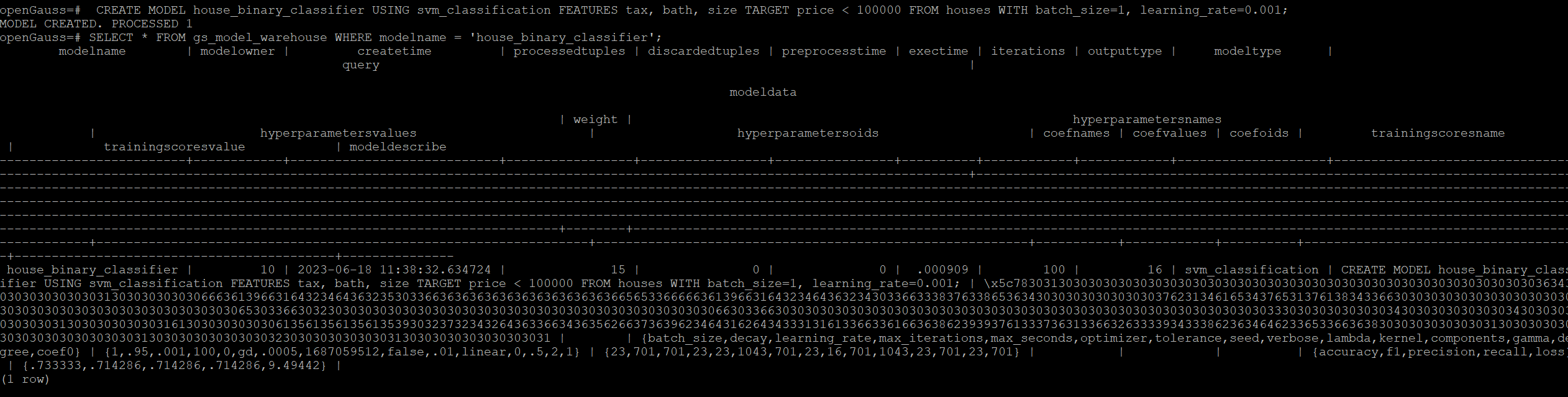
任务一：在gs\_model\_warehouse系统表中查看训练后的模型信息，将执行结果截图：

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



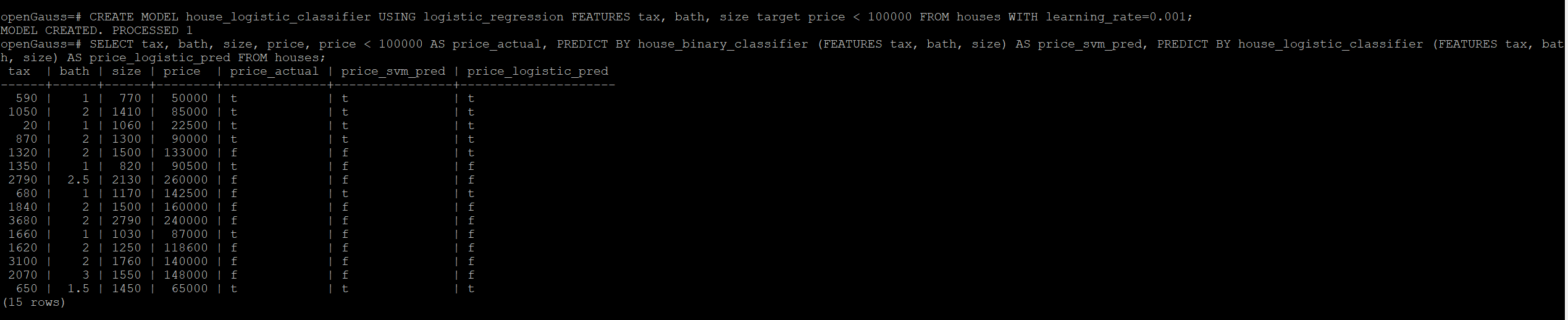
任务二：观察新模型的信息，将执行结果截图。

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？

分类模型和回归模型本质上是一样的。分类模型可将回归模型的输出离散化，回归模型也可将分类模型的输出连续化。

分类和回归的区别在于输出变量的类型。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。

实践思考题2：什么是SVM算法？

支持向量机（support vector machines, SVM）是一种二分类模型，它的基本模型是定义在特征空间上的间隔最大的线性分类器，间隔最大使它有别于感知机；SVM还包括核技巧，这使它成为实质上的非线性分类器。SVM的的学习策略就是间隔最大化，可形式化为一个求解凸二次规划的问题，也等价于正则化的合页损失函数的最小化问题。SVM的的学习算法就是求解凸二次规划的最优化算法。

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？

分类问题的常用评价指标有混淆矩阵（Confuse Matrix）、准确率（Accuracy）、精确率（Precision）、召回率（Recall）、F1 Score、P-R曲线（Precision-Recall Curve）、ROC、AUC等。

混淆矩阵：针对一个二分类问题，即将实例分成正类（positive）或负类（negative），在实际分类中会出现以下四种情况：真正类TP (True Positive )、假负类FN (False Negative )、假正类FP (False Positive )、真负类TN (True Negative )。混淆矩阵的每一行是样本的预测分类，每一列是样本的真实分类。

准确率：预测正确的样本数量占总量的百分比。

精确率：在模型预测为正样本的结果中，真正是正样本所占的百分比。

召回率：在实际为正样本中，被预测为正样本所占的百分比。

F1 Score：针对精准率和召回率都有其自己的缺点；如果阈值较高，那么精准率会高，但是会漏掉很多数据；如果阈值较低，召回率高，但是预测的会很不准确。F1 Score是一种调和平均数。

P-R曲线：描述精确率和召回率变化的曲线。

ROC曲线：ROC（Receiver Operating Characteristic）曲线，又称接受者操作特征曲线。曲线对应的纵坐标是TPR，横坐标是FPR。

AUC：AUC (Area Under Curve)是处于ROC曲线下方的那部分面积的大小。AUC越大，代表模型的性能越好。

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？

回归问题的常用评价指标有平均绝对误差（MAE）、均方误差（MSE）、均方根误差（RMSE）、可决系数（R-Squared）等。

平均绝对误差：这个指标是对绝对误差损失的预期值。

均方误差：这个指标是计算每一个样本的预测值与真实值差的平方，然后求和再取平均值。

均方根误差：这个指标是在均方误差的基础上再开方。

可决系数：可决系数R-Squared用于度量因变量的变异中可由自变量解释部分所占的比例，以此来判断统计模型的解释力。它将已解释的方差除以总方差，代表了总方差被预测变量所解释或决定的比率。决定系数R-Squared的值为0~1之间的值，R-Squared越接近于1，说明模型的效果越好，越接近于0，说明的模型效果越差。