openGauss AI特性创新实践课



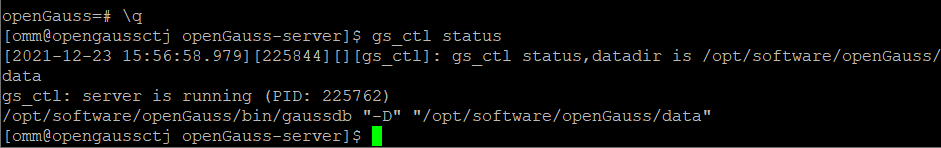
华为技术有限公司

# 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：

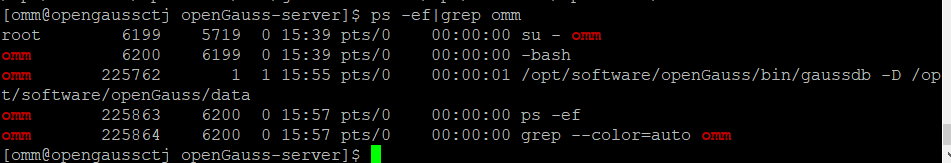
任务一：数据库状态验证

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？

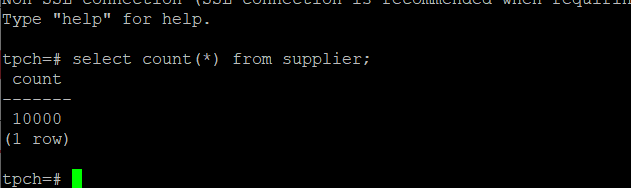
满足不同的运行平台，linux很多版本内核，二进制包所依赖的环境不一定能正常运行，所以大部分软件直接提供源码。方便定制，满足不同的需求。方便运维开发人员维护。

# 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：

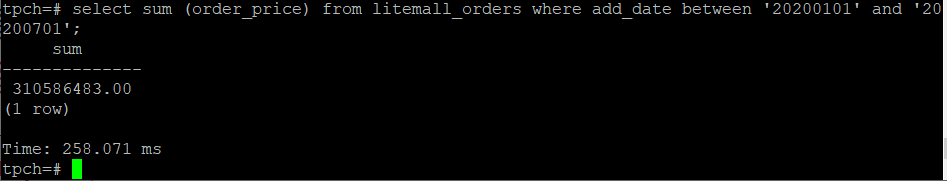
select count(\*) from supplier;;



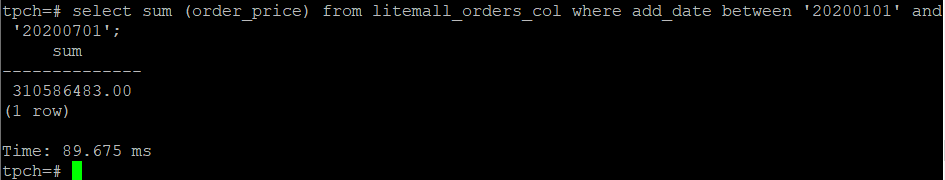
任务二：行存表与列存表执行效率对比

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';

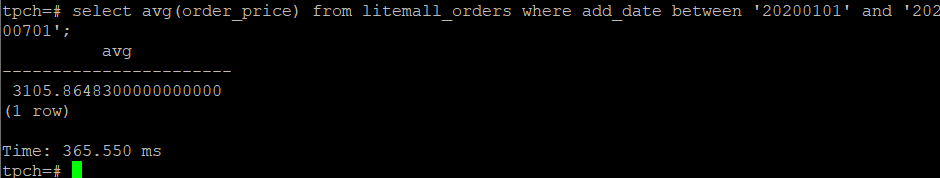


select sum (order\_price) from litemall\_orders\_col where add\_date between '20200101' and '20200701';

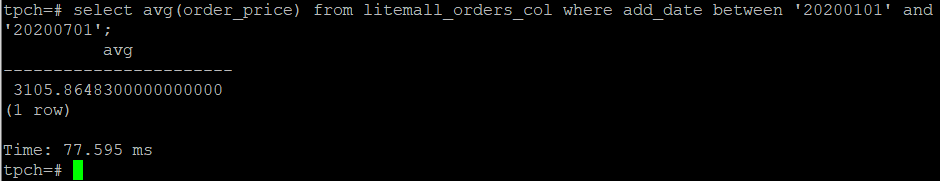


2. 2020年上半年litemall\_orders行存表与litemall\_orders\_col列存表中的order\_price的平均值查询，并对比执行效率截图

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

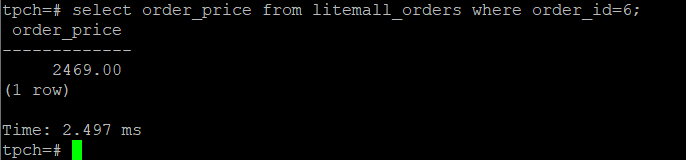


select avg (order\_price) from litemall\_orders\_col where add\_date between '20200101' and '20200701';

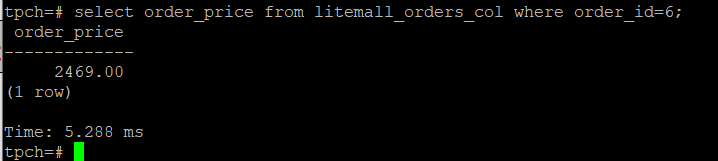


3. 查询litemall\_orders行存表与litemall\_orders\_col列存表中order\_id为6的order\_price的值，并对比执行效率截图。

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



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

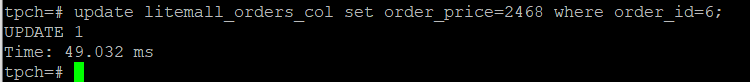


4. 将litemall\_orders行存表与litemall\_orders\_col列存表中order\_id为6的order\_price修改为2468，并对比执行效率截图。

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

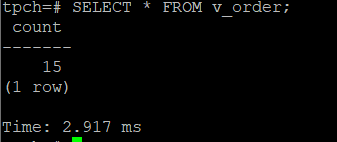
1. 创建物化视图所需要的表后，对表内容进行查询，对查询结果截图：

SELECT \* FROM test\_view;



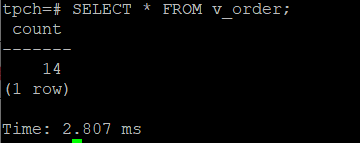
2. 使用物化视图统计人数，查询物化视图结果，将执行结果截图。

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;



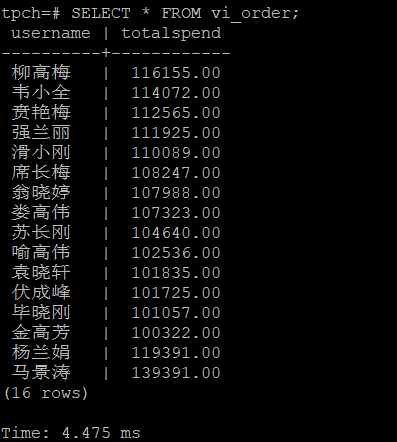
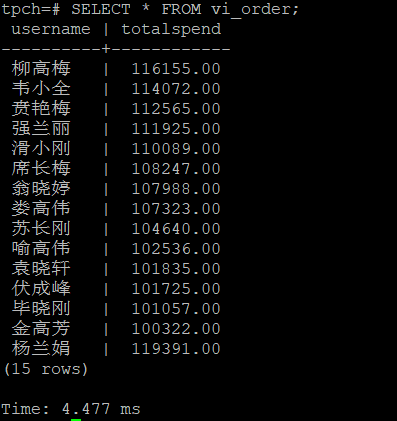
4. 创建增量物化视图，查询物化视图结果，将执行结果截图。

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？

行存表的写入是一次完成。如果这种写入建立在操作系统的文件系统上，可以保证写入过程的成功或者失败，数据的完整性因此可以确定。

列存表由于需要把一行记录拆分成单列保存，写入次数比行存表多，再加上磁头需要在盘片上移动和定位花费的时间，实际时间消耗会更大。所以，行存表在写入上占有很大的优势

行存表在数据记录（姓名、出生日期和电话号码）由多个字段组成且由某个键（在本例中为单调递增的ID）所唯一标识时效率更高，列存表在计算聚合的分析型工作负载，例如查找趋势、计算平均值等时效率更高。

实践思考题2：全量物化视图与增量物化视图有哪些差别？

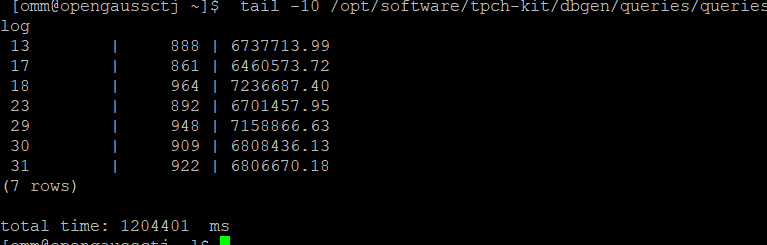
全量物化视图仅支持对创建好的物化视图做全量更新，而不支持做增量更新。创建全量物化视图语法和CREATE TABLE AS语法一致，不支持对全量物化视图指定NodeGroup创建。

# 关卡三、openGauss的AI4DB特性应用

任务一：使用X-Tuner进行参数优化

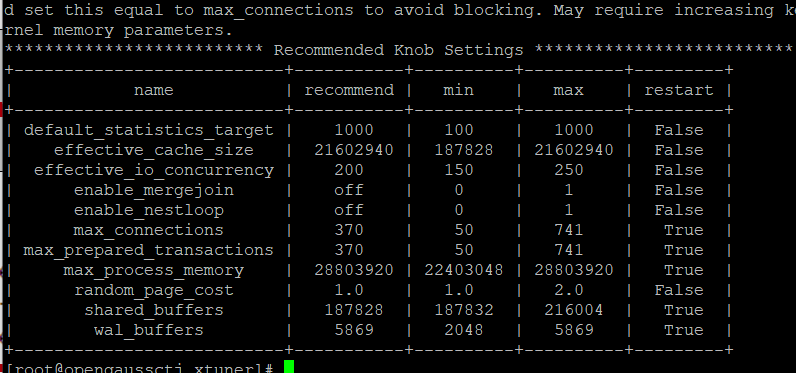
1. 执行TPCH脚本，获得测试时间，将执行结果截图：

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

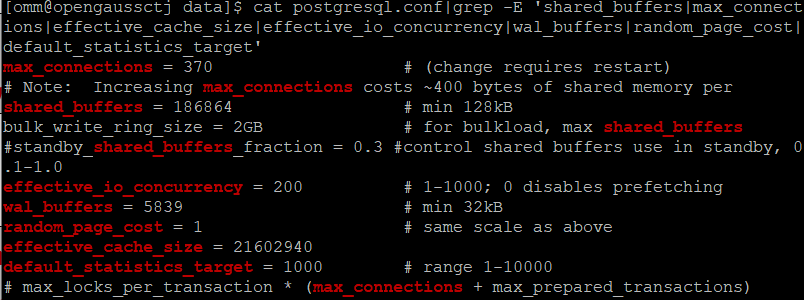
gs\_xtuner recommend --db-name tpch --db-user omm --port 5432 --host 127.0.0.1 --host-user omm



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

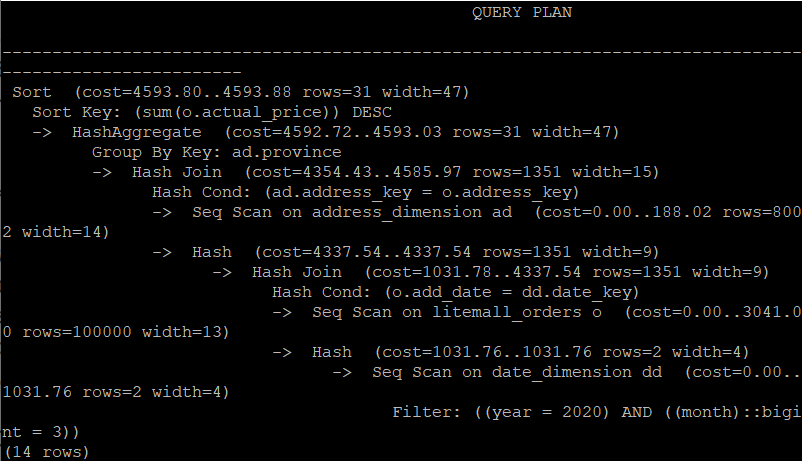
AND o.add\_date = dd.date\_key

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

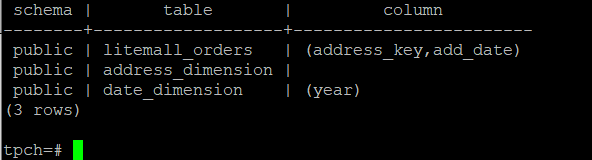
AND o.add\_date = dd.date\_key

AND dd.year = 2020

AND dd.month = 3

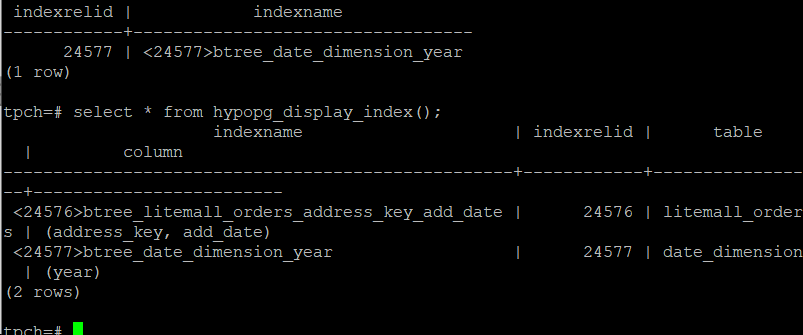
GROUP BY ad.province

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。

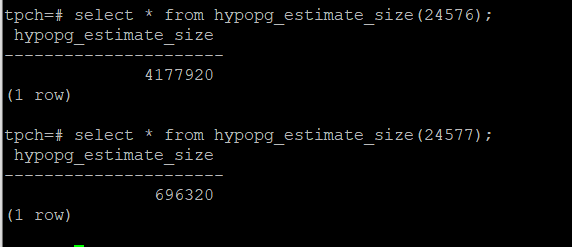
select \* from hypopg\_display\_index();



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

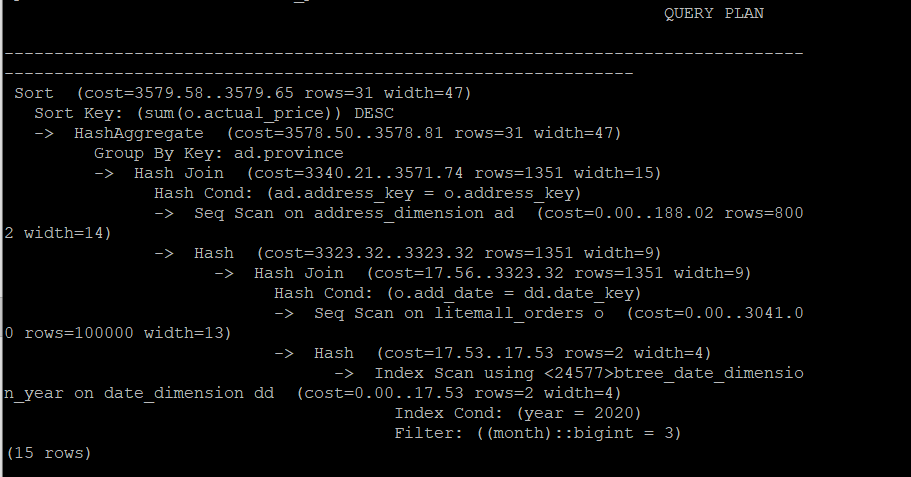
AND o.add\_date = dd.date\_key

AND dd.year = 2020

AND dd.month = 3

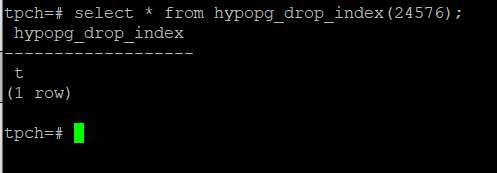
GROUP BY ad.province

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



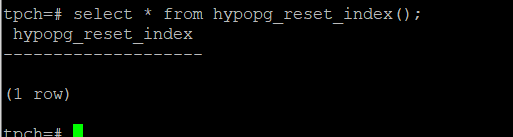
6. 删除某一个索引虚拟列，将执行结果截图。

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



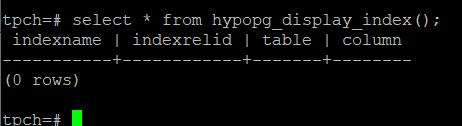
7. 删除某一个索引虚拟列，将执行结果截图。

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



8. 查看索引虚拟列，将执行结果截图。

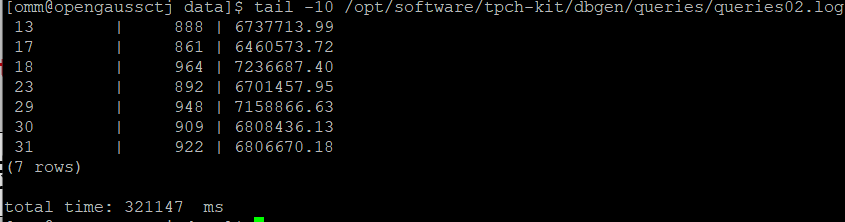
select \* from hypopg\_display\_index();



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？

**"shared\_buffers = 186864""max\_connections = 370""effective\_cache\_size = 21602940""effective\_io\_concurrency = 200""wal\_buffers = 5839""random\_page\_cost = 1""default\_statistics\_target = 1000"；减少搜索时间。**

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？

因为索引可以大大提高的性能。

第一通过唯一性索引可以保证数据库表中每一行数据的唯一性。

第二可以大大加快数据的检索速度。这也是创建索引最主要的原因。

第三可以加快表与表之间的连接。特别是在实现参考数据的完整性时特别有意义。

第四在使用分组和排序 子句进行数据检索时，同样可以显著减少查询中分组和排序的时间.

第五，通过使用索引，可以在查询的过程中，使用优化隐藏器，提高系统的性能。

1、调整数据结构的设计。这一部分在开发信息系统之前完成，程序员需要考虑是否使用ORACLE数据库的分区功能，对于经常访问的数据库表是否需要建立索引等。

2、调整应用程序结构设计。这一部分也是在开发信息系统之前完成，程序员在这一步需要考虑应用程序使用什么样的体系结构，是使用传统的Client、Server两层体系结构，还是使用Browser、Web、Database的三层体系结构。

3、调整数据库SQL语句。应用程序的执行最终将归结为数据库中的SQL语句执行，因此SQL语句的执行效率最终决定了ORACLE数据库的性能。

4、调整服务器内存分配。内存分配是在信息系统运行过程中优化配置的。

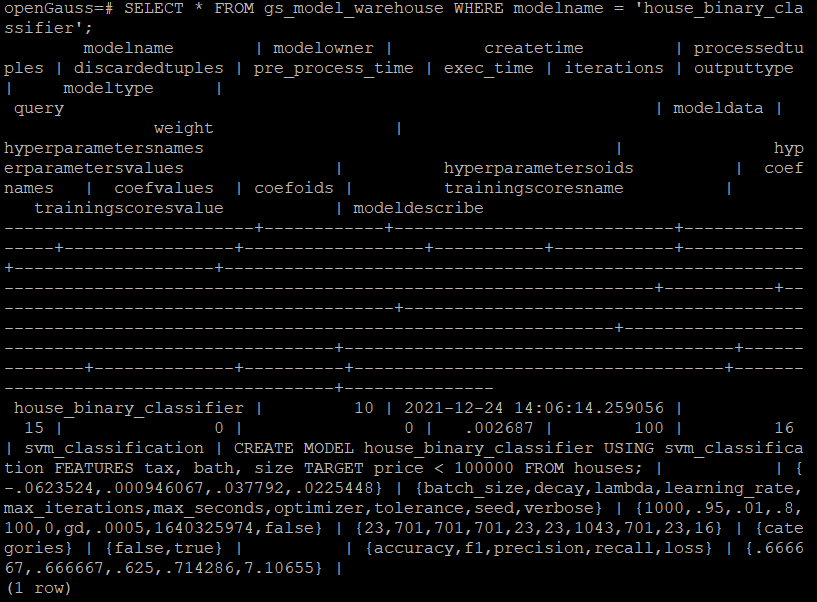
5、调整硬盘I/O，这一步是在信息系统开发之前完成的。数据库管理员可以将组成同一个表空间的数据文件放在不同的硬盘上，做到硬盘之间I/O负载均衡。

6、调整操作系统参数。

# 关卡四、openGauss的DB4AI特性应用

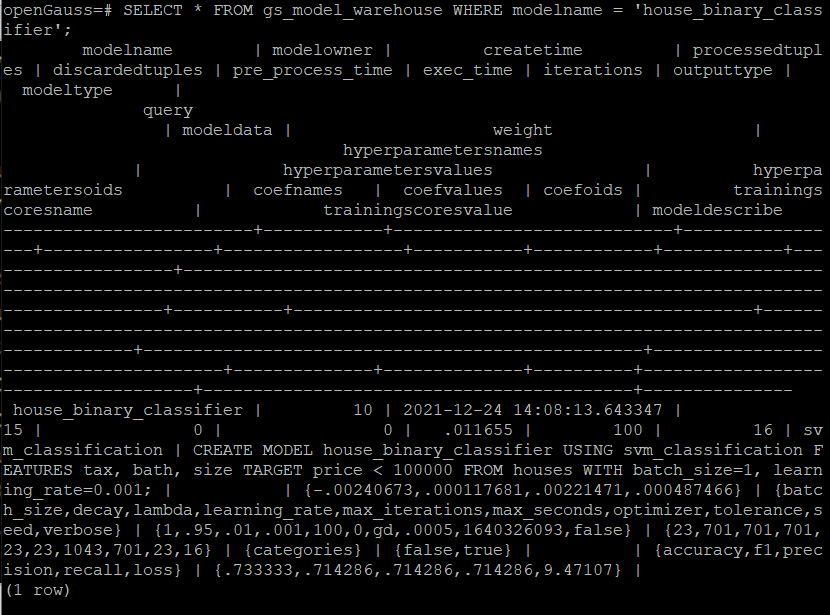
任务一：在gs\_model\_warehouse系统表中查看训练后的模型信息，将执行结果截图：

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



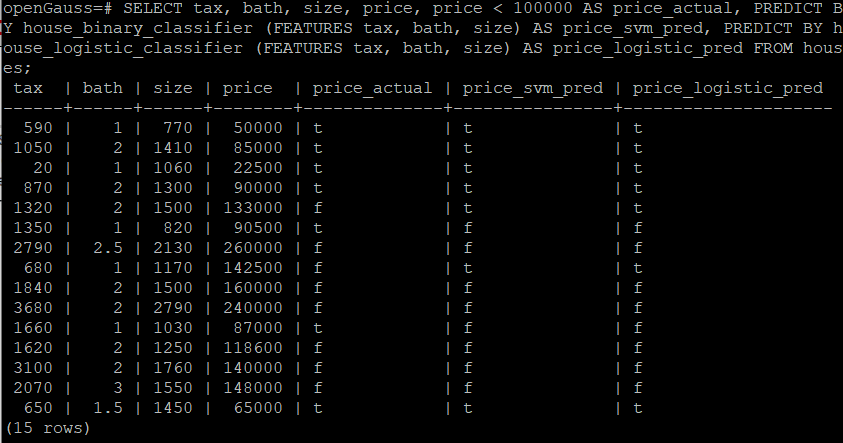
任务二：观察新模型的信息，将执行结果截图。

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？**

**（1）输出数据的类型**

**分类输出的数据类型是离散数据，也就是分类的标签。比如我们前面通过学生学习预测考试是否通过，这里的预测结果是考试通过，或者不通过，这2种离散数据。**

**回归输出的是连续数据类型。比如我们通过学习时间预测学生的考试分数，这里的预测结果分数，是连续数据。**

**2）第2个区别是我们想要通过机器学习算法得到什么？**

**分类算法得到是一个决策面，用于对数据集中的数据进行分类。**

**回归算法得到是一个最优拟合线，这个线条可以最好的接近数据集中的各个点。**

**3）第3个区别是对模型的评估指标不一样**

**在监督分类中，我们我们通常会使用正确率作为为指标，也就是预测结果中分类正确数据占总数据的比例**

**在回归中，我们用决定系数R平方来评估模型的好坏。R平方表示有多少百分比的y波动被回归线描述。**

**实践思考题2：什么是SVM算法？**

**SVM中文名为支持向量机，是常见的一种判别方法。在机器学习领域，是一个有监督的学习模型，通常用来进行模式识别、分类以及回归分析。**

**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？**

**一、accuracy**

**含义：在测试集上，预测对的样本占总样本的比例**

**二、precision**

**含义：在测试集上，预测对的正样本占总的预测为正样本的比例**

**三、recall**

**含义：在测试集上，预测对的正样本占总的本身标签为正样本的比例**

**四、f1-score**

**含义：在测试集上，模型对正负样本的综合预测能力**

**五、ROC图**

**含义：在测试集上，比较不同模型的性能**

**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？**

1. **SSE**

**含义：误差平方和**

**二、R-square**

**含义：决定系数**

**三、Adjusted R-Square**

**含义：校正决定系数**