openGauss AI特性创新实践课



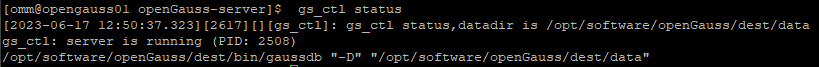
华为技术有限公司

# 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：

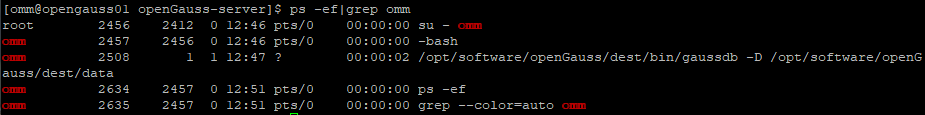
任务一：数据库状态验证

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？

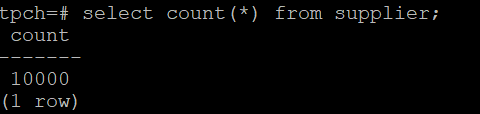
源码安装的优点，编译安装过程，可以设定参数，按照需求，进行安装，并且安装的版本，可以自己选择，灵活性比较大。

# 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';

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：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;

图片包含 图示

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;

图片包含 图示

描述已自动生成图片包含 图示

描述已自动生成

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？

在执行相同的SQL语句时，行存表与列存表查询的方式不同，所以执行的时间不同。在只需要涉及到的列时，列存表效率更高。在需要多列数据时，行存表效率更高。

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。

[增量物化视图](https://www.modb.pro/db/525469#section1980464963719)：可以对物化视图增量刷新，需要用户手动执行语句完成对物化视图在一段时间内的增量数据刷新。与全量创建物化视图不同在于目前增量物化视图所支持场景较小，目前物化视图创建语句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

文本

描述已自动生成

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

图片包含 图形用户界面

描述已自动生成

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'

电脑萤幕的截图

描述已自动生成

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;

文本

描述已自动生成

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

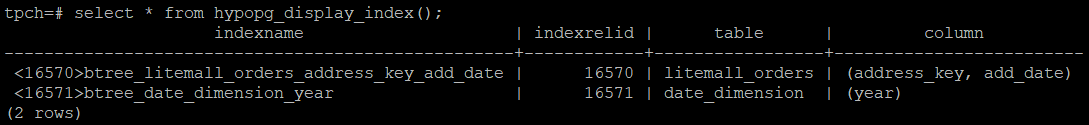
ORDER BY SUM(o.actual\_price) DESC');

文本

描述已自动生成

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);

文本

描述已自动生成文本

描述已自动生成

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;

文本

描述已自动生成

6. 删除某一个索引虚拟列，将执行结果截图。

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

文本

描述已自动生成

7. 删除某一个索引虚拟列，将执行结果截图。

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

文本

描述已自动生成

8. 查看索引虚拟列，将执行结果截图。

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

图片包含 日程表

描述已自动生成

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

文本

描述已自动生成

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？

对shared\_buffers，max\_connections，effective\_cache\_size，effective\_io\_concurrency，wal\_buffers，random\_page\_cost，default\_statistics\_target 参数进行了优化。增加了每次的最大连接数，加快了模型的训练。

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？

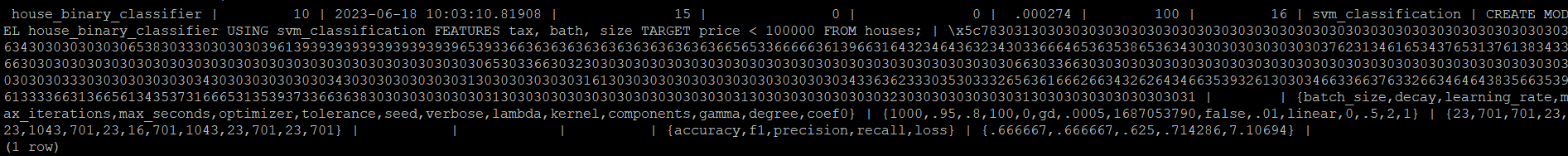
通过创建唯一性索引，可以保证数据库表中每一行数据的唯一性。可以大大加快数据的检索速度。在使用分组和排序子句进行数据检索时，同样可以显著减少查询中分组和排序的时间。

优化方法：查询语句优化，优化子查询，优化数据库参数，缓存。

# 关卡四、openGauss的DB4AI特性应用

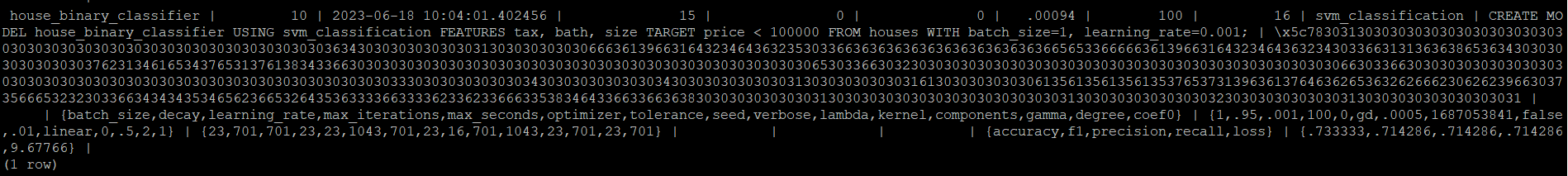
任务一：在gs\_model\_warehouse系统表中查看训练后的模型信息，将执行结果截图：

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;

文本

描述已自动生成

实践思考题1：分类模型与回归模型有何不同？

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？

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？

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：  
（1）若一个实例是正类，并且被预测为正类，即为真正类TP(True Positive )  
（2）若一个实例是正类，但是被预测为负类，即为假负类FN(False Negative )  
（3）若一个实例是负类，但是被预测为正类，即为假正类FP(False Positive )  
（4）若一个实例是负类，并且被预测为负类，即为真负类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。

表格

中度可信度描述已自动生成

准确率(Accuracy)：

预测正确的样本数量占总量的百分比：

手机屏幕的截图

描述已自动生成

精准率(Precision)：

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

文本

描述已自动生成

召回率(Recall)：

在实际为正样本中，被预测为正样本所占的百分比。

文本

描述已自动生成

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？

平均绝对值误差（MAE）：

均绝对值误差就是计算每一个样本的预测值和真实值的差的绝对值，然后求和再取平均值。

手机屏幕截图

中度可信度描述已自动生成

均方误差(MSE)：

均方误差就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。

手机屏幕截图

中度可信度描述已自动生成

均方根误差（RMSE）

均方根误差就是在均方误差的基础上再开方。

手机屏幕截图

中度可信度描述已自动生成