openGauss AI特性创新实践课



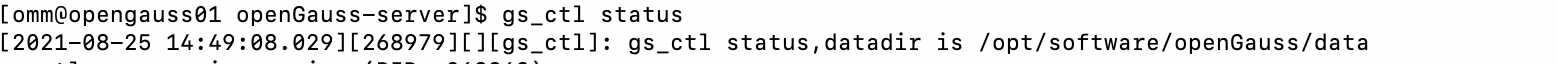
华为技术有限公司

# 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）

图片包含 图示

描述已自动生成

实验思考题：为什么需要通过源码编译，安装数据库？

由于系统是Linux的用户可操作性较高，通过编译安装的方法能实现对很多部分自定义安装，对数据库整体也有充分了解。

# 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;;

文本

描述已自动生成

任务二：行存表与列存表执行效率对比

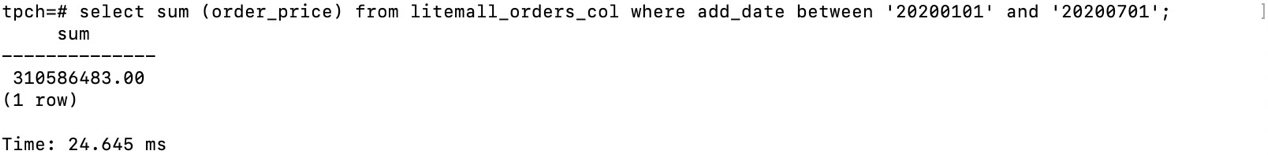
1. 2020年上半年litemall\_orders行存表与litemall\_orders\_col列存表中的order\_price的总和查询，并对比执行效率截图

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';



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

中度可信度描述已自动生成

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

图片包含 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

低可信度描述已自动生成

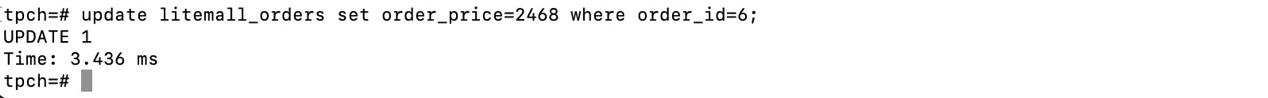
select order\_price from litemall\_orders\_col where order\_id=6;

文本

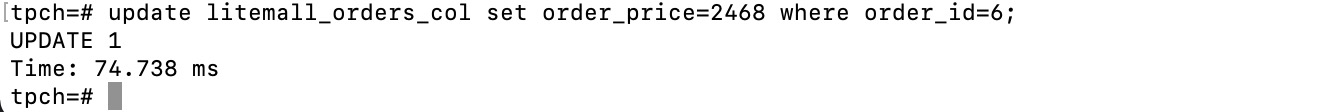
中度可信度描述已自动生成

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;



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;

手机屏幕截图

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

低可信度描述已自动生成

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;

图片包含 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;

表格

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;

表格

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？

1. 因为存储方式不同，在进行查询时首先查找的key-value不同。在发现当进行全局或一个范围内查询时列存储的更高。

实践思考题2：全量物化视图与增量物化视图有哪些差别？

1. 全量物化视图和增量物化视图都能加速查询时间，区别是全量物化视图更适合进行全局查询，增量中插入等更方便

# 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

图形用户界面, 文本

中度可信度描述已自动生成

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

表格

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'

文本

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, 信件

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();

表格

描述已自动生成

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);

手机屏幕截图

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, 信件

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);

图片包含 应用程序

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();

图片包含 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();

手机屏幕截图

中度可信度描述已自动生成

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

图片包含 表格

描述已自动生成

文本

中度可信度描述已自动生成

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

图片包含 表格

描述已自动生成

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？

一个是共享区域，最大连接数，有效cache容量、有效io并行数、还有wal\_buffers random\_page\_cost = 1 default\_statistics\_target。通过修改一方面提高运行效率，另一方面各项吞吐率提高。

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？

索引就像书本中的目录标签，他可以大大提高查询时的速度。

优化有几种：1、sql语句的执行计划是否正常。

2、减少应用和数据库的交互次数、同一个sql语句的执行次数。

3、数据库实体的碎片的整理（特别是对某些表经常进行insert和delete动作，尤其注意，索引字段为系列字段、自增长字段、时间字段，对于业务比较频繁的系统，最好一个月重建一次）。 4、减少表之间的关联，特别对于批量数据处理，尽量单表查询数据，统一在内存中进行逻辑处理，减少数据库压力。

5、对访问频繁的数据，充分利用数据库cache和应用的缓存。

6、数据量比较大的，在设计过程中，为了减少其他表的关联，增加一些冗余字段，提高查询性能。

# 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';

表格

描述已自动生成

任务二：观察新模型的信息，将执行结果截图。

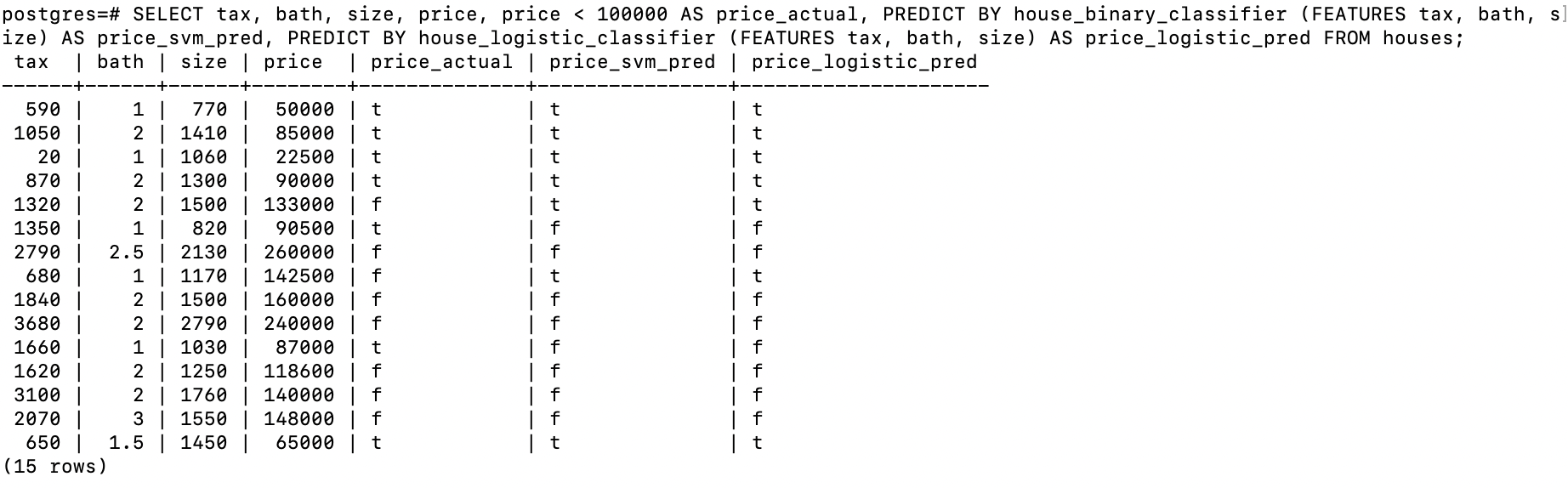
postgres=# SELECT \* FROM gs\_model\_warehouse WHERE modelname = 'house\_binary\_classifier';

表格

描述已自动生成

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？

分类模式是有具体的类别以及类别信息也就是离散的，而回归模型本质上可以认为是一段连续的值。它们都属于监督学习。

实践思考题2：什么是SVM算法？

SVM的全称是Support Vector Machine，即支持向量机，主要用于解决模式识别领域中的数据分类问题，属于有监督学习算法的一种。SVM要解决的问题可以用一个经典的二分类问题加以描述。

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？

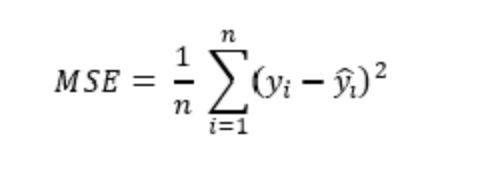
第一个是准确率是我们常见的评价指标之一，一般定义是，分类正确的样本数占总样本的比例数。

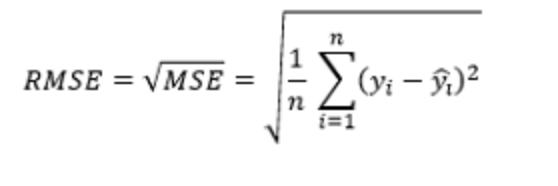
第二种是精准率和召回率。精准率可以解释为，预测为正例的样本中，有多少是真的正例，召回率可以解释为，真实的正例的样本中，有多少被预测出来

第三种是AUC是另一种评价二分类算法的指标，被定义为 ROC 曲线下的面积。

第四种还有混淆矩阵

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？

第一个是MSE均方误差 

第二个是RMSE 

第三个是MAE平均绝对误差图示, 示意图

描述已自动生成