openGauss AI特性创新实践课



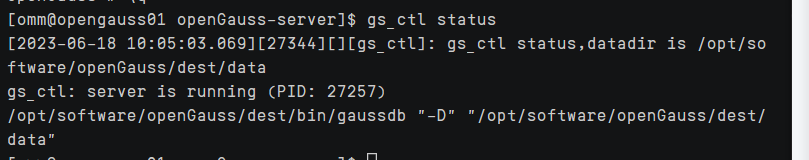
华为技术有限公司

# 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：

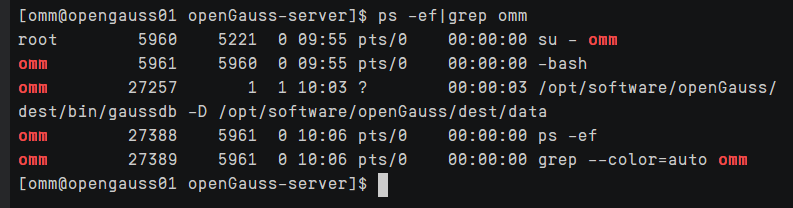
任务一：数据库状态验证

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？

编译安装数据库的主要原因有以下几点：

1.定制化需求：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能力。源代码允许对数据库进行自定义配置和优化，以满足特定的需求。可以根据硬件配置、操作系统版本和其他因素进行优化，以获得更好的性能和安全性。

2.最新功能和修复：数据库项目通常会在其源代码中发布最新的功能增强和错误修复。如您通过源码编译安装数据库，您将能够及时获得这些更新，并享受到最新功能和改进。

3.安全性和稳定性：编译安装数据库还可以增强系统的安全性和稳定性。可以选择特定的安全选项、加密算法和身份验证机制来保护数据库。此外，通过源码编译还可以避免依赖于第三方软件包的安全漏洞，因为可以控制编译时使用的软件版本。

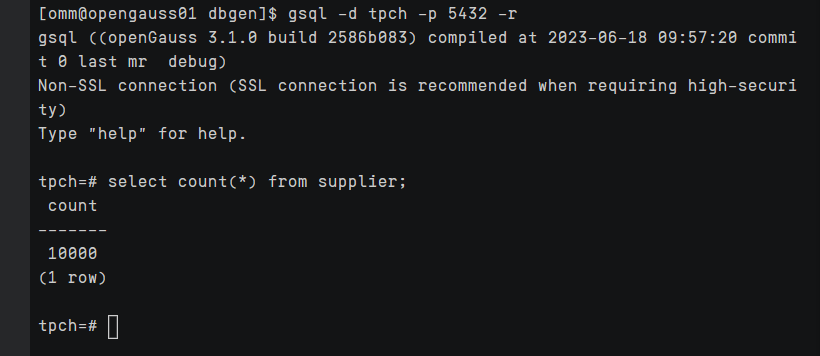
4.学习和调试：对于开发人员和系统管理员来说，通过编译和安装数据库的源码可以提供更好的学习和调试环境。可以深入研究数据库的内部工作原理，并在需要时对其进行调试和修改。

# 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：

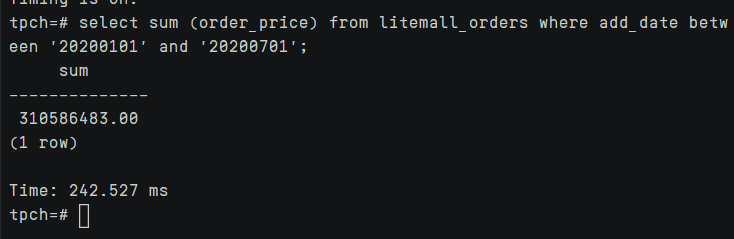
select count(\*) from supplier;;



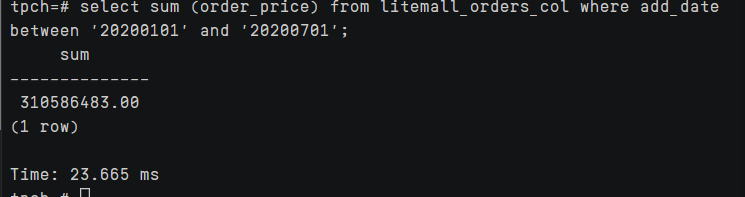
任务二：行存表与列存表执行效率对比

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';

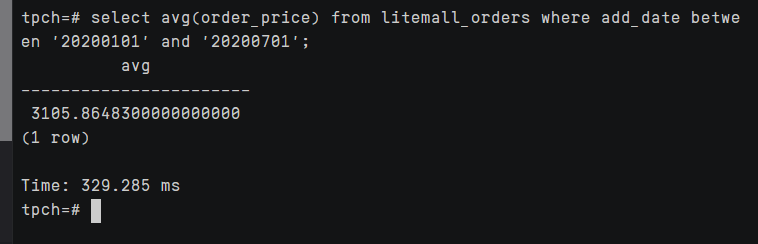


select sum (order\_price) from litemall\_orders\_col where add\_date between '20200101' and '20200701';

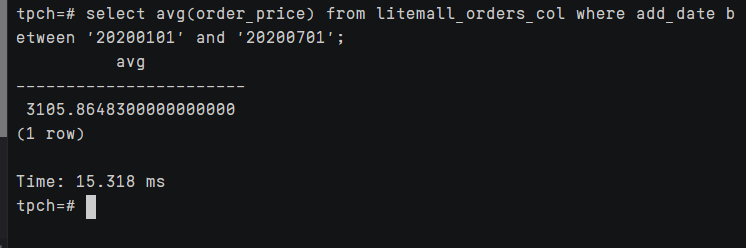


2. 2020年上半年litemall\_orders行存表与litemall\_orders\_col列存表中的order\_price的平均值查询，并对比执行效率截图

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

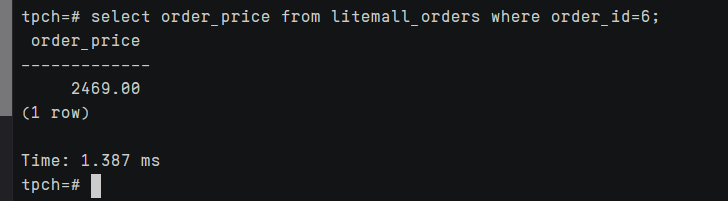


select avg (order\_price) from litemall\_orders\_col where add\_date between '20200101' and '20200701';

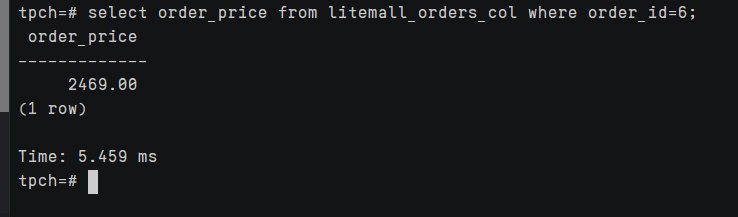


3. 查询litemall\_orders行存表与litemall\_orders\_col列存表中order\_id为6的order\_price的值，并对比执行效率截图。

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

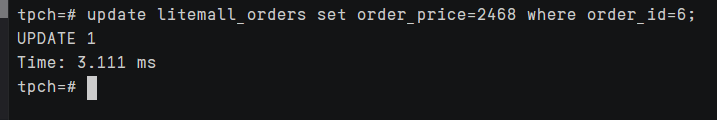


select order\_price from litemall\_orders\_col where order\_id=6;

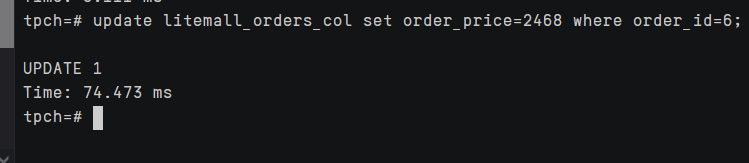


4. 将litemall\_orders行存表与litemall\_orders\_col列存表中order\_id为6的order\_price修改为2468，并对比执行效率截图。

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

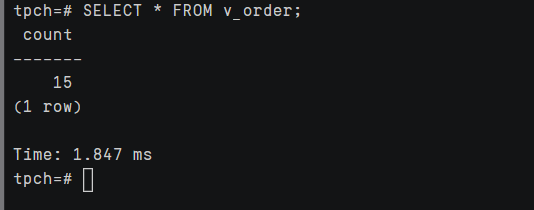
1. 创建物化视图所需要的表后，对表内容进行查询，对查询结果截图：

SELECT \* FROM test\_view;



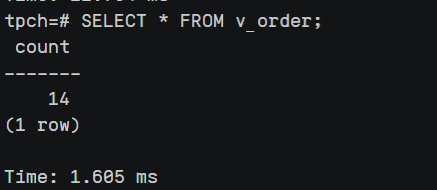
2. 使用物化视图统计人数，查询物化视图结果，将执行结果截图。

SELECT \* FROM v\_order;



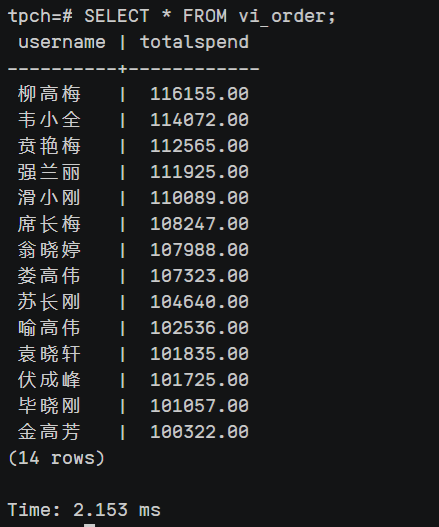
3. 对表进行操作后，刷新物化视图，查询物化视图结果，将执行结果截图。

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？

行存表以行为单位存储数据，即将一行完整的数据存储在一起。这种存储方式适合于对整行数据进行读取和更新的操作。当执行涉及到多个列的查询或需要更新大部分或全部列的操作时，行存表效率较高。这是因为对于行存表来说，读取一整行数据或更新一整行数据时，可以直接读取或写入连续的内存块，减少了磁盘 I/O 操作和数据的解析成本。

列存表则将数据按列进行存储，即将同一列的数据存储在一起。这种存储方式适合于分析型查询，即需要对大量行进行聚合、筛选或统计的操作。当执行涉及到少量列的查询或需要对某些列进行聚合计算的操作时，列存表效率较高。这是因为列存表中的数据压缩和编码方式针对列数据的特点进行了优化，可以减少磁盘 I/O 操作和内存消耗，并提供更高的查询性能。

具体来说，行存表在以下类型的 SQL 操作中效率较高：①单行读取：当需要读取一整行数据时，行存表可以更快地读取整个数据块。②更新操作：当需要更新大部分或全部列的数据时，行存表可以直接写入整个数据块，减少写入开销。③事务处理：行存表适合于事务处理，因为事务通常需要读取和更新整行数据。

列存表在以下类型的 SQL 操作中效率较高：①聚合查询：当需要对大量行进行聚合计算、统计或筛选操作时，列存表可以仅加载所需的列数据，减少数据传输和解析的开销。②分析型查询：当需要分析大量数据的趋势、模式或统计特征时，列存表可以提供更高的查询性能和压缩率。③列选择性较高的查询：当查询仅涉及少量列，并且列的选择性较高（即某些列的值相对较少）时，列存表可以快速跳过不符合条件的列数据，减少不必要的 I/O 操作和数据解析。

需要注意的是，具体的效率差异还受到数据库管理系统的实现方式、硬件配置、数据量等因素的影响。

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（Materialized View）和增量物化视图（Incremental Materialized View）是在数据仓库和数据库系统中常见的两种物化视图类型，它们有以下几个主要差别：

①数据存储方式：

全量物化视图：全量物化视图包含完整的预计算数据，即它存储了所有数据的副本。当底层数据发生变化时，需要通过刷新（refresh）操作重新计算和更新物化视图的内容。

增量物化视图：增量物化视图只存储增量数据，即它仅存储与底层数据发生变化的部分。增量物化视图通过维护和追踪变化的数据，可以更高效地更新和保持与底层数据的一致性。

②刷新机制：

全量物化视图：全量物化视图的刷新操作通常需要重新计算和加载整个视图的数据。这可以通过将视图与底层数据进行比较并更新不一致的部分来实现。刷新过程可能需要较长时间，特别是当底层数据量较大或计算复杂度较高时。

增量物化视图：增量物化视图使用增量更新的方式进行刷新。它可以通过捕获和记录底层数据的变化（如插入、更新、删除操作）来追踪增量。当刷新增量物化视图时，只需要应用最新的变化到物化视图中，而不需要重新计算整个视图。

③查询性能：

全量物化视图：全量物化视图在刷新之后可以提供实时的完整数据，因此对于涉及整个数据集的查询，可以获得较高的性能提升。然而，刷新操作可能会导致一些延迟，并且在数据量大或计算复杂的情况下，可能会影响到整体查询性能。

增量物化视图：增量物化视图的刷新操作通常比全量物化视图更快，因为只需处理变化的数据。对于只涉及增量数据的查询，增量物化视图可以提供更快的查询性能。但对于查询需要使用到全量数据的情况，增量物化视图可能需要通过合并增量数据和历史数据来提供完整的结果，因此在这种情况下性能可能较差。

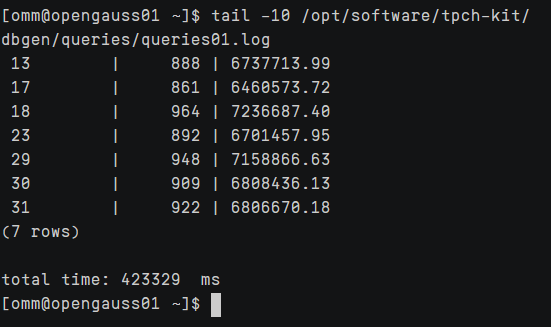
综上所述，全量物化视图适用于需要实时完整数据的场景，而增量物化视图适用于需要高效增量更新和查询的场景。选择使用哪种物化视图类型需要考虑数据量、数据变化频率、查询模式和性能要求等因素。

# 关卡三、openGauss的AI4DB特性应用

任务一：使用X-Tuner进行参数优化

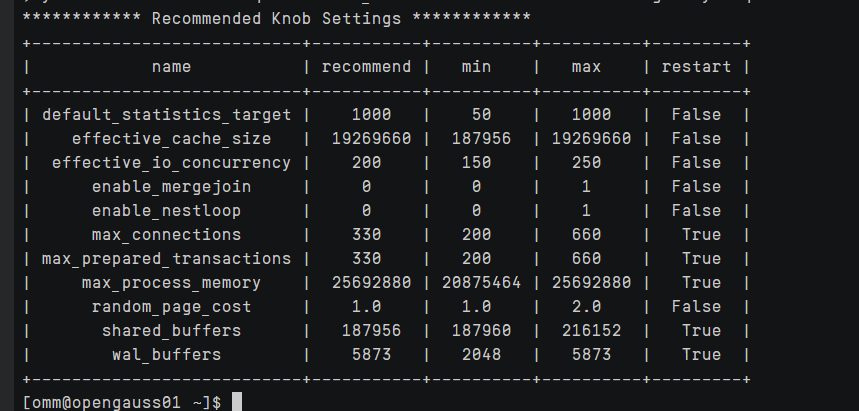
1. 执行TPCH脚本，获得测试时间，将执行结果截图：

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

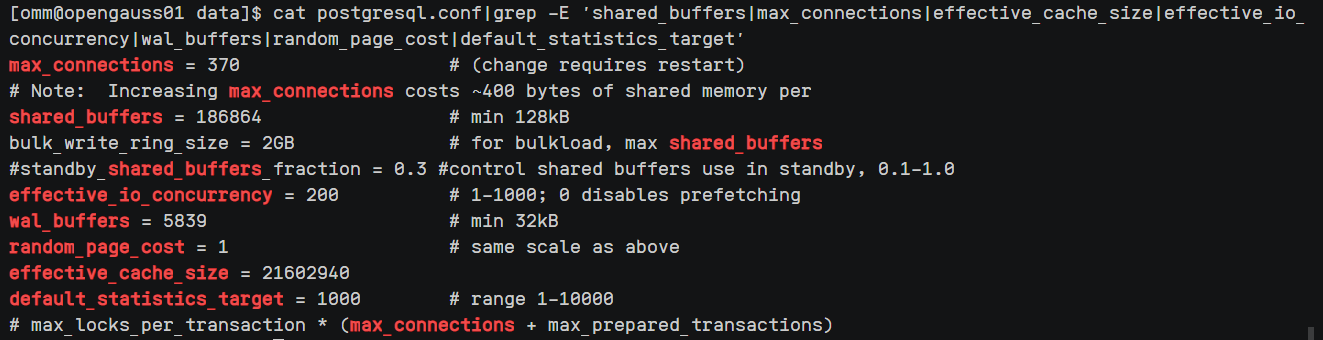
gs\_xtuner recommend --db-name tpch --db-user omm --port 5432 --host 127.0.0.1 --host-user omm



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

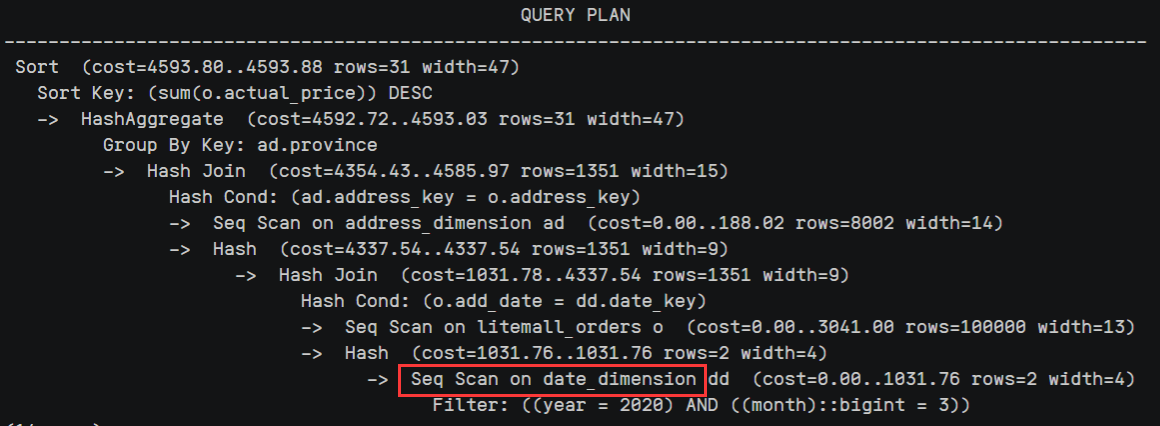
AND o.add\_date = dd.date\_key

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

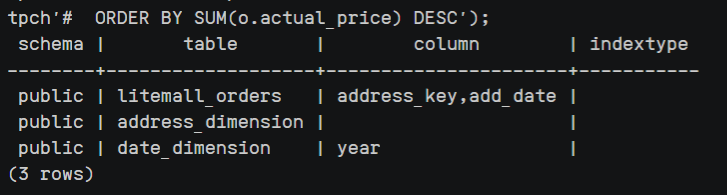
AND o.add\_date = dd.date\_key

AND dd.year = 2020

AND dd.month = 3

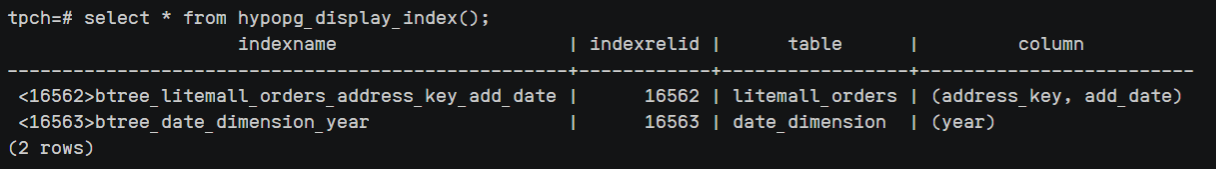
GROUP BY ad.province

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。

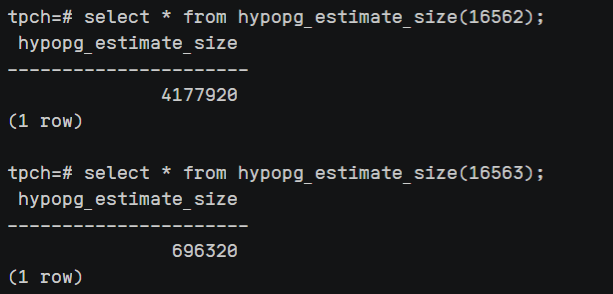
select \* from hypopg\_display\_index();



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

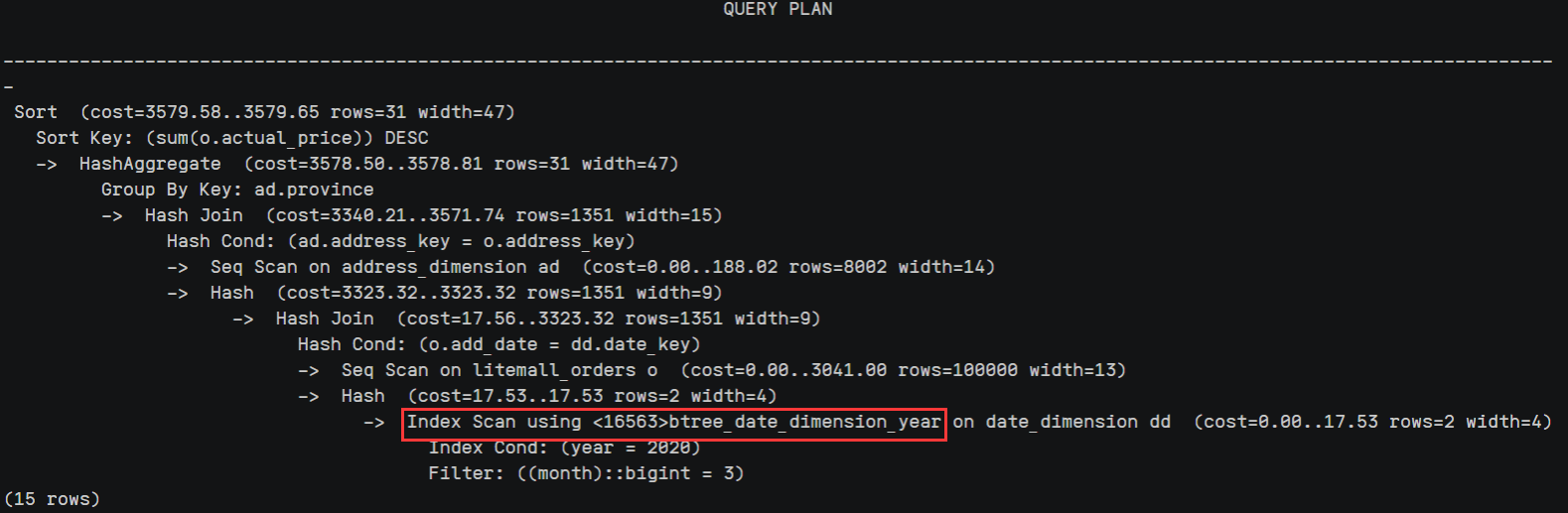
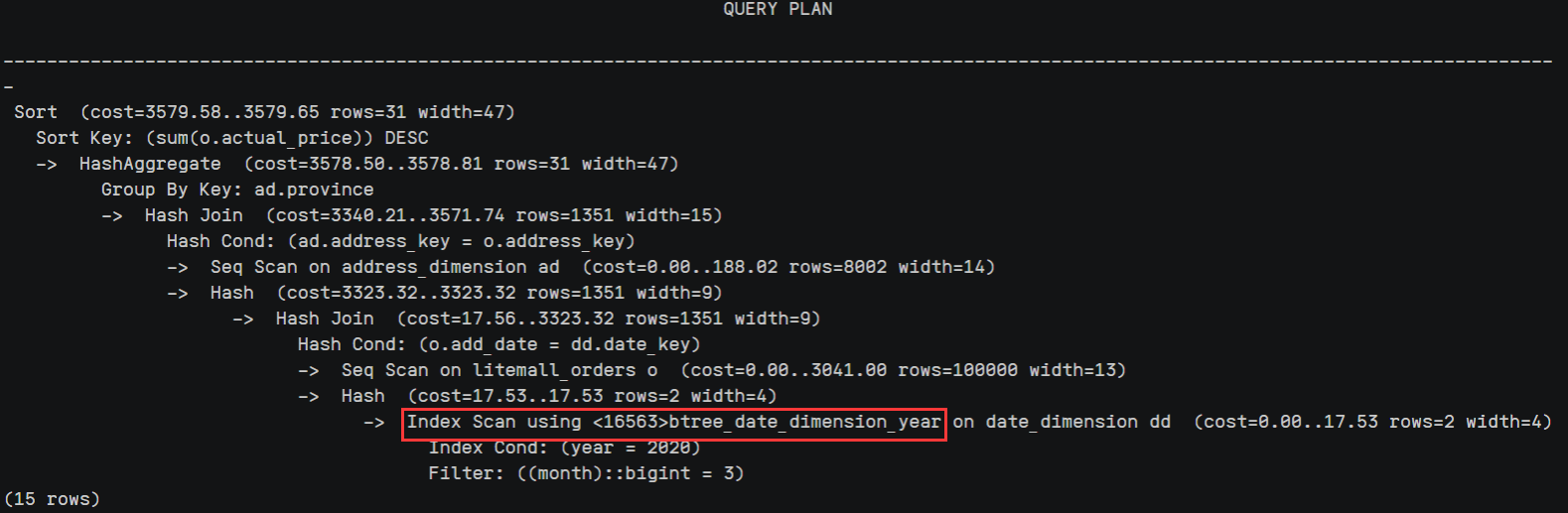
AND o.add\_date = dd.date\_key

AND dd.year = 2020

AND dd.month = 3

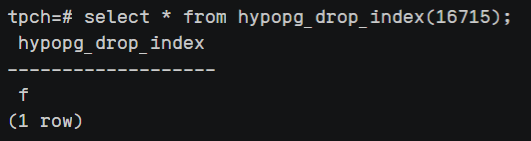
GROUP BY ad.province

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



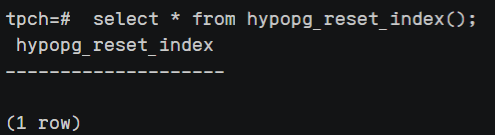
6. 删除某一个索引虚拟列，将执行结果截图。

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



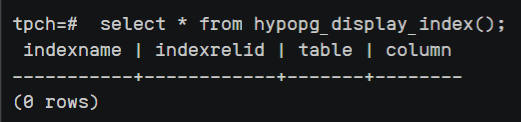
7. 删除某一个索引虚拟列，将执行结果截图。

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



8. 查看索引虚拟列，将执行结果截图。

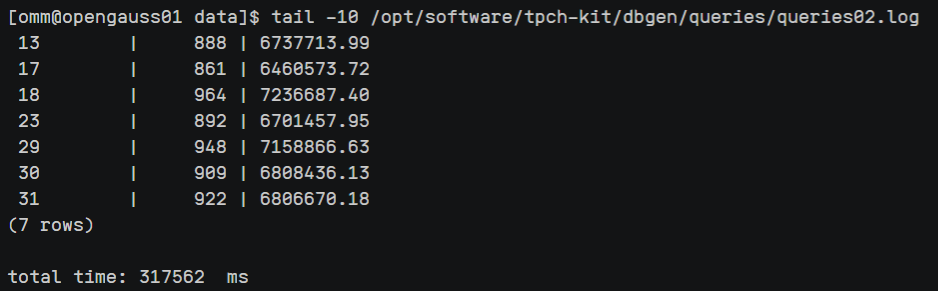
select \* from hypopg\_display\_index();



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：

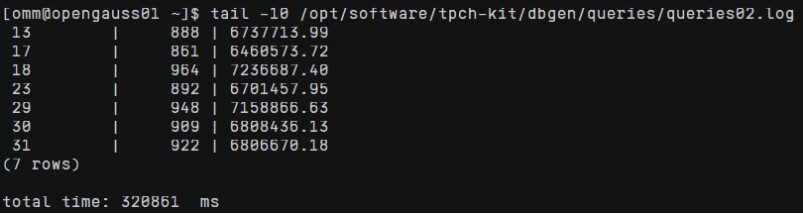
gsql -d tpch -p 5432 -r -f /opt/software/tpch-kit/dbgen/queries/queries.sql > /opt/software/tpch-kit/dbgen/queries/queries02.log



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？

针对执行的queries.sql进行优化，为customer、order、part表增加主键索引，在order表的o\_orderdate列，以及lineitem表的l\_shipdate列上创建索引

改善模型性能：通过调整参数值，可以提高模型在某个性能指标上的表现。例如，在机器学习中，通过调整模型的超参数（如学习率、正则化参数等），可以提高模型的准确性或泛化能力。

提高效率：优化参数可以使模型更快地收敛或更高效地执行。通过合适的参数配置，可以减少模型训练或推断所需的时间和计算资源。

适应不同的数据和场景：不同的数据集和应用场景可能需要不同的参数配置。通过优化参数，可以使模型具有更好的适应性，能够在不同的数据和场景下表现出更好的性能。

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？

提高查询性能：索引可以加快查询的速度。通过在表上创建索引，数据库可以更快地定位到包含所需数据的位置，从而减少了扫描整个表的开销。

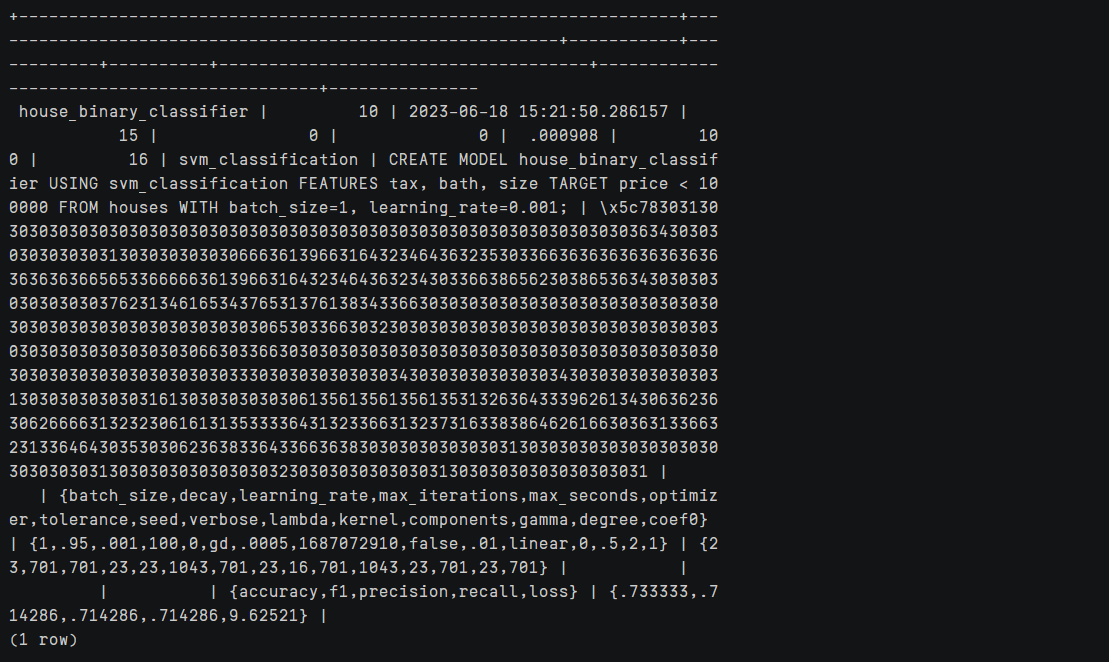
减少 I/O 操作：索引可以减少磁盘 I/O 操作的次数。当查询使用索引时，数据库可以直接访问索引所在的数据页，而不需要读取整个数据表，从而减少了磁盘 I/O 的开销。

优化连接操作：对于连接操作（如 JOIN），索引可以加速连接过程，减少连接的时间复杂度。通过在连接列上创建索引，数据库可以更快地匹配连接条件，提高连接操作的效率。

# 关卡四、openGauss的DB4AI特性应用

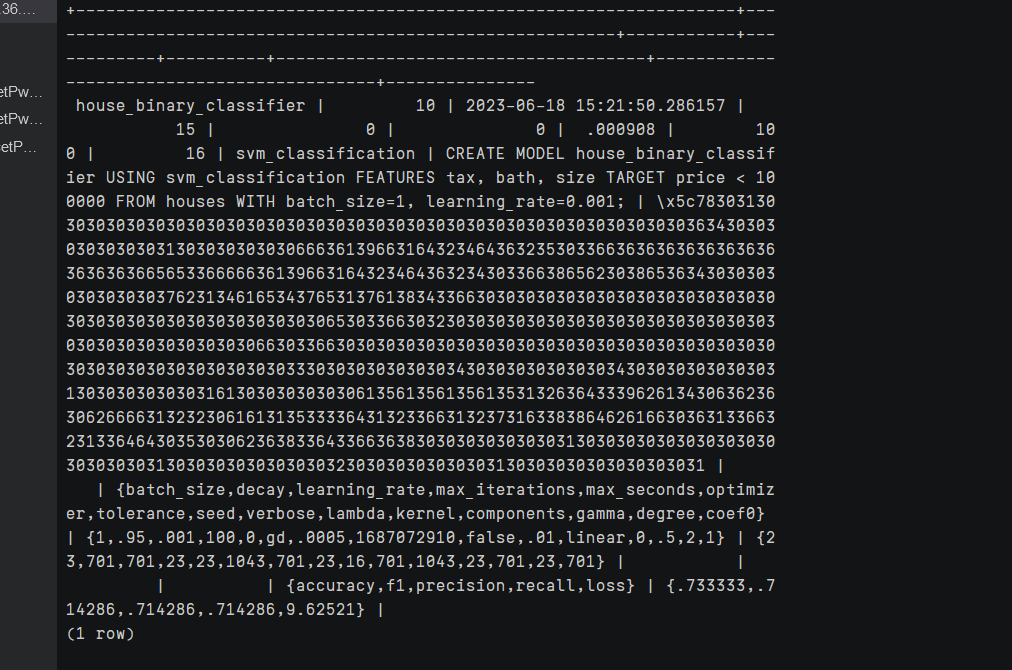
任务一：在gs\_model\_warehouse系统表中查看训练后的模型信息，将执行结果截图：

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



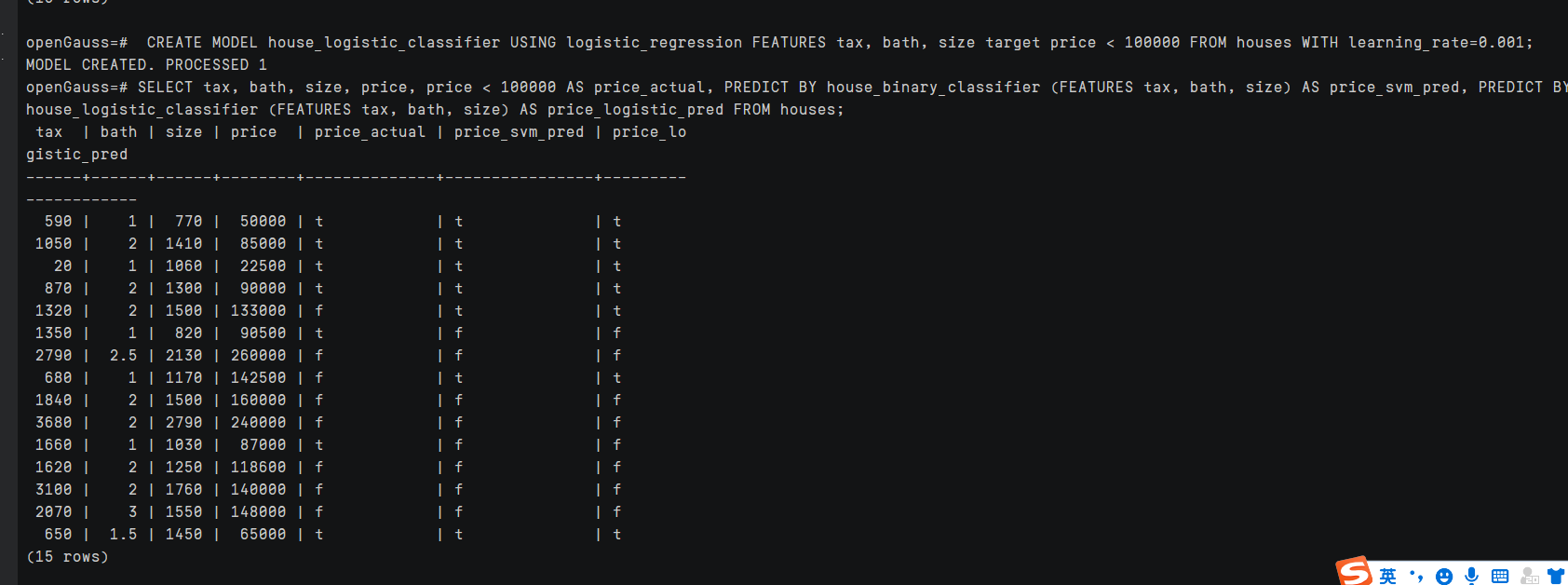
任务二：观察新模型的信息，将执行结果截图。

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是机器学习中两种不同类型的模型，用于解决不同类型的问题。

分类模型用于将输入数据分为不同的类别或标签。它们用于预测离散的输出变量，例如确定一张图片中的物体是猫还是狗，预测一个邮件是垃圾邮件还是非垃圾邮件，或者将一个人的疾病分为患有糖尿病或没有糖尿病等。分类模型的输出通常是一组离散的标签或类别之一。

回归模型用于预测连续的输出变量。它们通过分析输入数据中的模式和关联性来预测一个或多个连续的数值。例如，预测房屋的价格，基于特征如房间数量、面积和位置等；预测销售量，基于广告支出和市场规模等因素。回归模型的输出是一个或多个实数值，而不是离散的类别。

在算法层面上，分类和回归模型之间的区别通常体现在模型的目标函数和预测输出的处理方式上。分类模型通常使用分类算法，如逻辑回归、决策树、支持向量机（SVM）和神经网络等。这些算法通过最大化分类的准确性或最小化分类误差来进行训练和优化。回归模型则使用回归算法，如线性回归、岭回归、决策树回归和支持向量回归等。这些算法通过拟合输入特征和输出变量之间的函数关系来进行训练和优化。

总结起来，分类模型用于预测离散的类别或标签，而回归模型用于预测连续的数值。它们在目标函数和处理方式上有所不同，使用不同的算法来进行训练和预测。

实践思考题2：什么是SVM算法？

SVM（支持向量机，Support Vector Machine）是一种常用的监督学习算法，可用于分类和回归任务。SVM 的主要目标是找到一个最佳的超平面（在二维空间中为一条直线，在多维空间中为一个超平面），用于将不同类别的数据样本分隔开。

SVM 的基本思想是将数据映射到高维空间中，在该空间中找到一个超平面，使得不同类别的样本尽可能地分开，并且离超平面最近的数据点距离最大。这些距离最近的数据点被称为支持向量。SVM通过最大化支持向量到超平面的间隔来提高分类的准确性，从而具有较好的泛化能力。

在分类任务中，SVM算法可以应用于线性可分和线性不可分的情况。对于线性可分的情况，SVM使用线性超平面将不同类别的样本分隔开。对于线性不可分的情况，SVM利用核函数将数据映射到高维空间中，使得数据在高维空间中线性可分。

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？

1.准确率（Accuracy）：准确率是最常用的分类评价指标之一，表示模型正确预测的样本数与总样本数之比。准确率越高，表示模型的分类预测越准确。

准确率 = (预测正确的样本数) / (总样本数)

尽管准确率是一个重要指标，但在某些情况下，它可能会受到样本不平衡的影响，特别是当不同类别的样本数量差异较大时。

2.精确率（Precision）：精确率是指在所有被模型预测为正例的样本中，实际为正例的比例。精确率衡量了模型在预测为正例的样本中的准确性。

精确率 = (真阳性) / (真阳性 + 假阳性)

精确率高表示模型能够将负例判断为负例，避免误判。

3.召回率（Recall）：召回率是指在所有实际为正例的样本中，被模型预测为正例的比例。召回率衡量了模型对正例样本的覆盖程度。

召回率 = (真阳性) / (真阳性 + 假阴性)

召回率高表示模型能够捕捉到更多的真实正例，减少漏判。

4.F1值（F1-Score）：F1值是精确率和召回率的调和平均值，综合考虑了模型的准确性和覆盖度。F1值越高，表示模型的综合性能越好。

F1值 = 2 \* (精确率 \* 召回率) / (精确率 + 召回率)

F1值在样本不平衡的情况下更加稳健，因为它同时考虑了精确率和召回率。

5.特定类别的评价指标：针对多类别分类问题，可以计算每个类别的精确率、召回率和F1值，以了解模型在不同类别上的性能。

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？

平均绝对误差（Mean Absolute Error，MAE）：MAE是预测值与实际值之间差异的平均绝对值。它衡量了模型预测结果的平均偏差大小，数值越小表示模型的预测越准确。

MAE = (1/n) \* Σ|预测值 - 实际值|

均方误差（Mean Squared Error，MSE）：MSE是预测值与实际值之间差异的平方的平均值。MSE在计算过程中平方了误差，因此对较大误差给予了更高的惩罚。

MSE = (1/n) \* Σ(预测值 - 实际值)^2

MSE的数值越小表示模型的预测越准确。

均方根误差（Root Mean Squared Error，RMSE）：RMSE是MSE的平方根，它使用与实际值相同的单位度量误差。与MSE相比，RMSE的值更接近实际值的范围，容易解释。

RMSE = √(MSE)

R²（R-squared）：R²衡量了模型对目标变量的解释能力。它表示模型解释的方差占总方差的比例，数值范围从0到1。R²等于1表示模型能够完全解释目标变量的变异性，而R²等于0表示模型无法解释目标变量的变异性。

R² = 1 - (Σ(预测值 - 实际值)^2) / (Σ(实际值 - 平均值)^2)

R²的值越接近1表示模型的拟合程度越好，越接近0表示模型的拟合程度越差。

这些评价指标可以帮助我们评估回归模型的性能。选择合适的指标取决于具体的问题和需求。通常，MAE、MSE和RMSE用于度量预测误差的大小，而R²用于衡量模型的解释能力。