|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 《数据库原理》实验报告 | | | | |
| 题目：实验四  视图与索引 | 学号 | 姓名 | 班级 | 日期 |
| **2021302111** | **禹泽海** | **JKC012101** | **2023/10/20** |

**一 .头歌平台截图**



头歌平台实验四完成截图

**二 .实验内容、步骤以及结果（线下实验）**

**题目1 .**基于以上的索引（Course表：cno的主键索引，cname的普通索引；Student表：sno的主键索引，IX\_ngd复合索引），用explain得到的查询计划观察每个查询语句中索引的使用情况。

注意：本练习除了在头歌平台调用explain，还需要将执行结果截图，并在实验报告中，说明具体的自己的分析情况。

①explain select \* from course;

②explain select \* from course where cno = ‘1’;

③explain select \* from course where cname=’数据库’ ;

④explain select \* from course where cname like ‘%数据库%’;

⑤explain select \* from course where cname like ‘数据库%’;

⑥explain select \* from student where sname ='张立' and sno='2001';

⑦explain select \* from student where sname ='张立' and sgender='男' and sdept='IS';

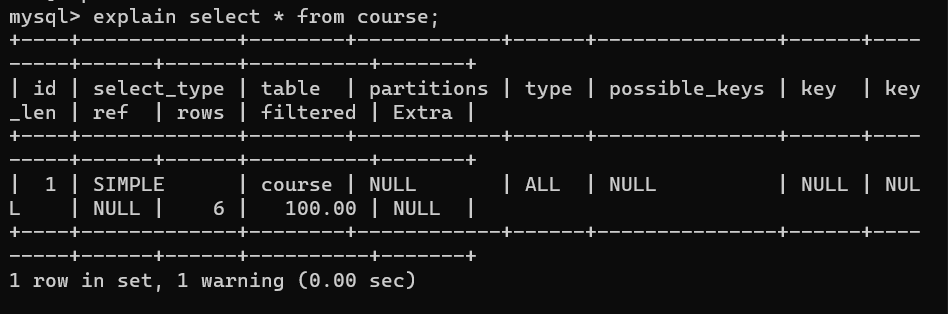
⑧explain select \* from student where sname ='张立' and sgender='男';

⑨explain select \* from student where sname ='张立';

⑩explain select \* from studentwhere sgender ='男';

⑪explain select \* from student where sgender ='男' and sdept='IS';

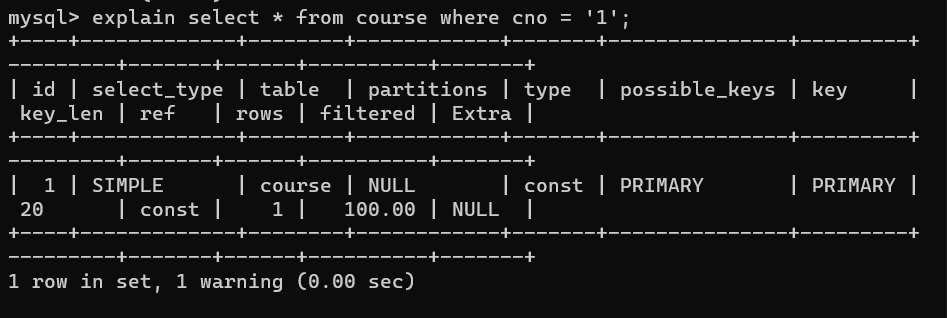
实验过程：



这是一个简单的SELECT语句，没有特定的过滤条件。

可能会使用全表扫描，因为没有指定任何筛选条件，除非有特定的默认索引。

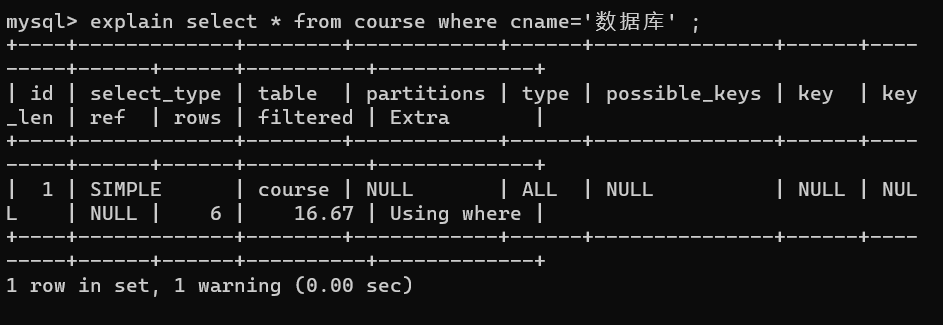
由于没有提供详细的表结构和索引信息，无法确定是否使用了索引。



这个查询使用了 cno 列的等值条件来筛选数据。

如果 cno 列有适当的索引，可能会使用该索引进行快速查找。

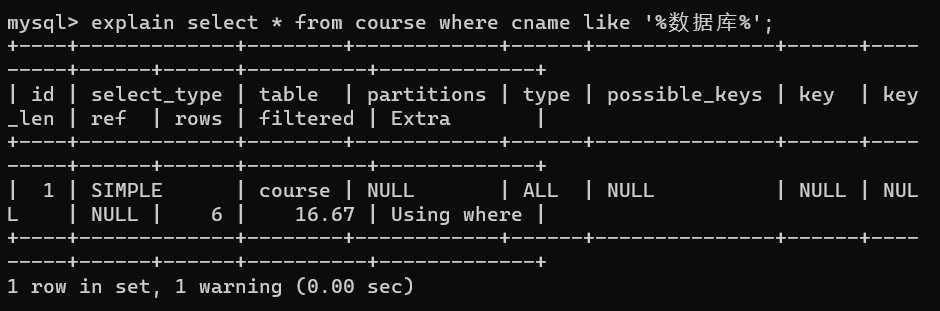
查询的性能可能会取决于表结构和索引情况。



这个查询使用了 cname 列的等值条件来筛选数据。

如果 cname 列有适当的索引，可能会使用该索引进行快速查找。

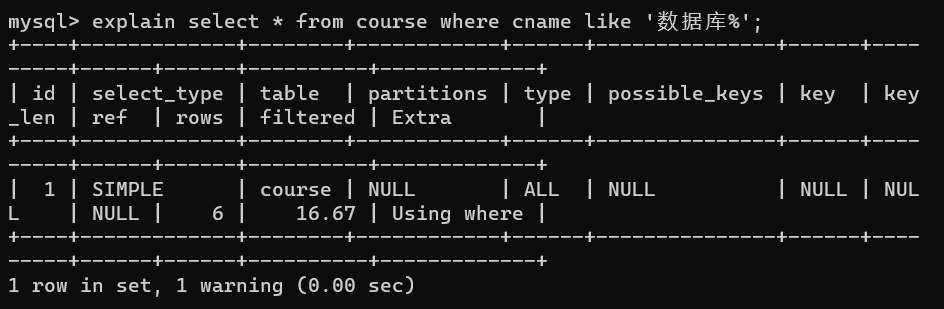
查询的性能可能会取决于表结构和索引情况。



这个查询使用了 cname 列的模糊查询，查找包含 "数据库" 的任何位置的数据。

通常情况下，这种模糊查询不会使用索引，可能会进行全表扫描。

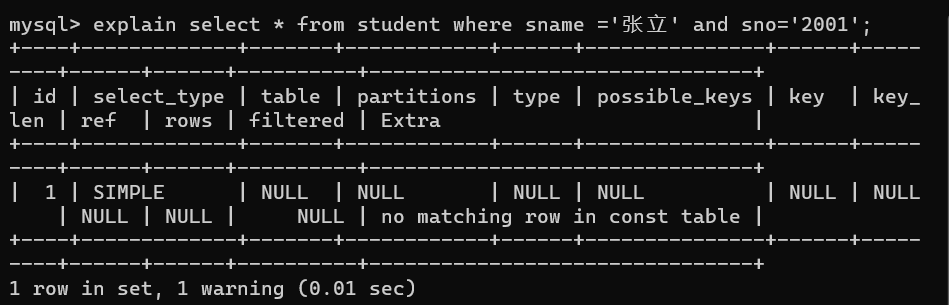
查询性能可能较差，特别是在大型数据集上。



这个查询使用了 cname 列的模糊查询，查找以 "数据库" 开头的数据。

也许不会使用索引，可能进行全表扫描。

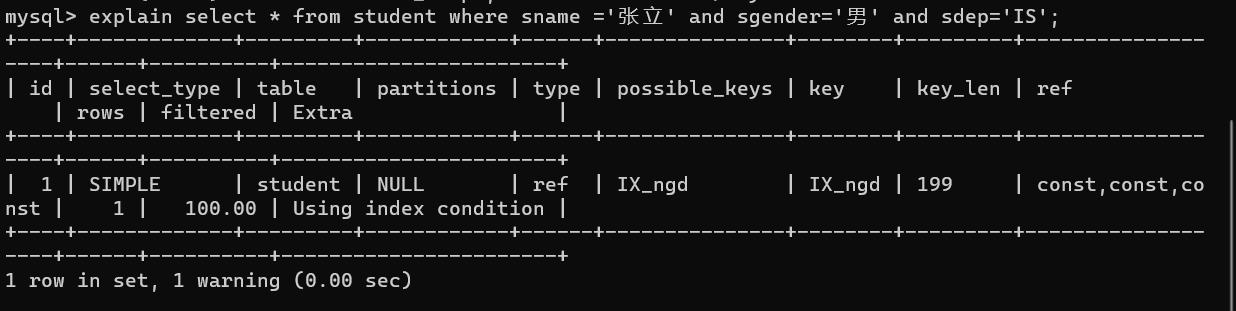
查询性能可能取决于表的大小。



这个查询使用了 sname 和 sno 列的等值条件。

如果这两列都有适当的索引，可能会使用这些索引进行查找。

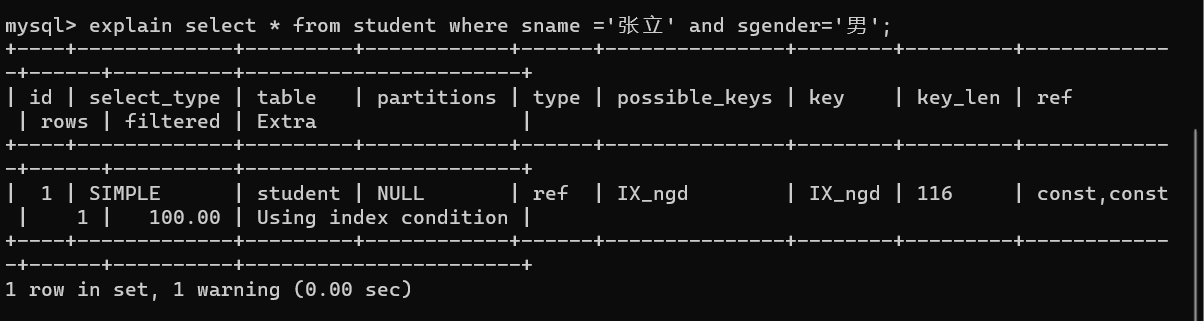
查询性能可能会较好，特别是如果这些列是唯一键或主键。



这个查询使用了 sname、sgender 和 sdept 列的等值条件。

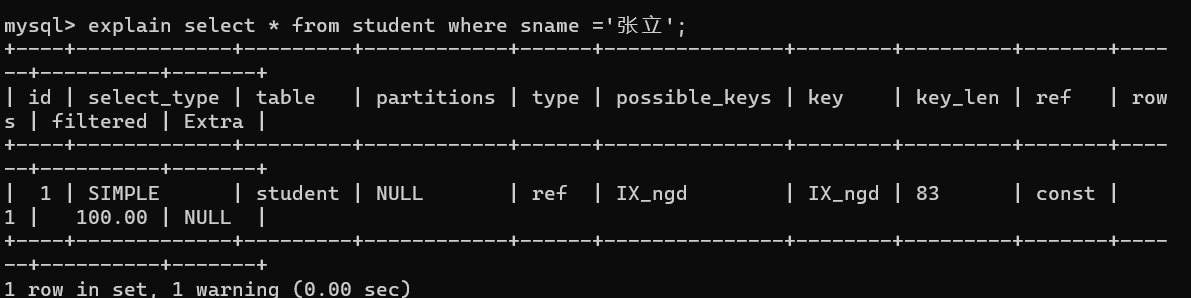
如果这些列都有适当的索引，可能会使用这些索引进行查找。

查询性能可能会较好，特别是如果这些列是唯一键或主键。



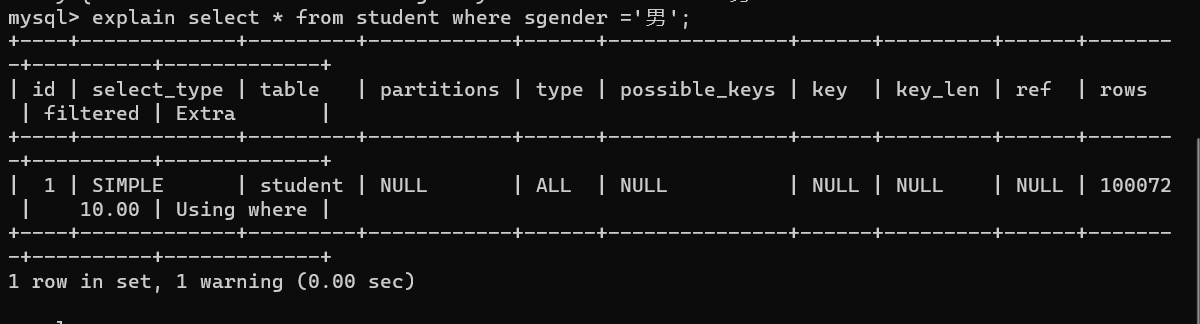
这个查询使用了 sname 和 sgender 列的等值条件。

查询性能可能会依赖于是否存在索引，并且如果有适当的索引，性能可能会较好。



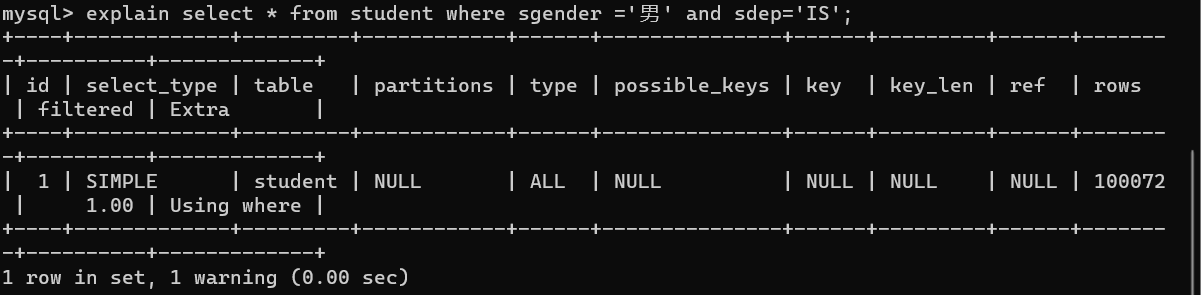
这个查询使用了 sname 列的等值条件。

查询性能可能会依赖于是否存在索引，如果有适当的索引，性能可能会较好。



这个查询使用了 sgender 列的等值条件。

查询性能可能会依赖于是否存在索引，如果有适当的索引，性能可能会较好。



这个查询使用了 sgender 和 sdept 列的等值条件。

查询性能可能会依赖于是否存在索引，如果有适当的索引，性能可能会较好。

**题目2 .**

以下实验内容，涉及构造大表（可能需要修改MySQL默认参数设置），请连接本地MySQL服务器完成。

假设有一个如下的基本表userinfo，自己设计一个实验验证索引对数据库查询效率的提升作用。（40分）

create table userinfo

(

user\_id int primary key, //用户ID

username varchar(10), //用户名

gender char(1), //性别

age int, //年龄

c\_id int //学院编号

)

(1)验证有索引和无索引的查询效率差异。

(2)验证单字段窄索引和多字段构成的宽索引的查询效率异，注意理解宽索引中的最左匹配原则。

(3)验证聚集索引（主键索引）与二级索引（非主键索引）的查询效率差异：对同一个字段分别建立聚集索引与非聚集索引，比较查询效率。（选做）

(4)目前，MySQL只有memory引擎同时支持B树索引与Hash索引。在MySQL中创建基于memory存储引擎的基本表，基于该表验证B树索引与Hash索引的查询效率差异。（选做）

提示与要求：

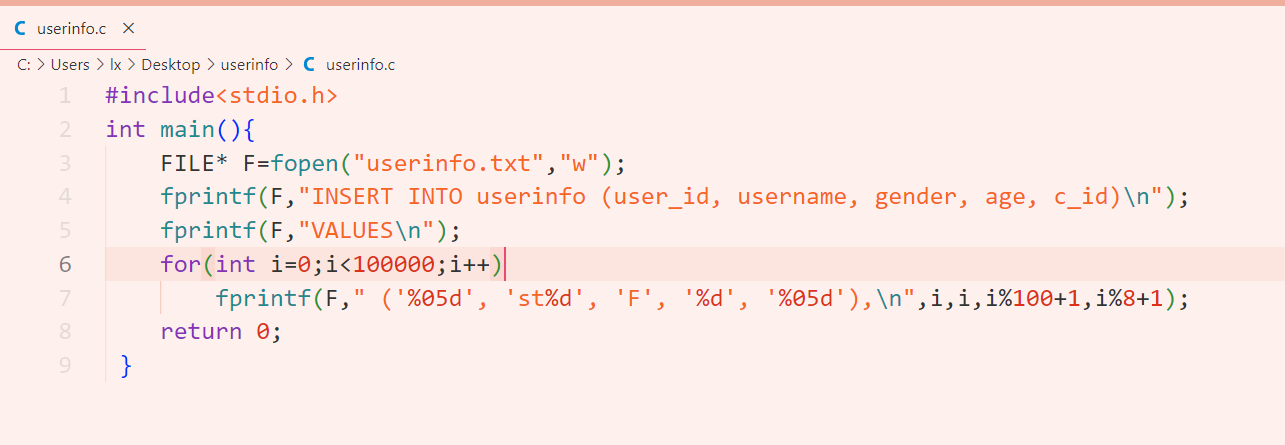
(1)导入大量数据（至少在10W条以上，根据使用机器的性能差异，也可能需要更多条数据）在验证中使用。

(2)用EXPLAIN命令分析SQL查询是否使用了所创建的索引，基于此进行性能原因分析。

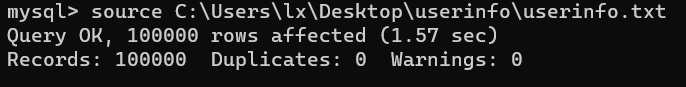
(3)通常情况下，B树索引适用于范围查询，Hash索引适用于点查询。

实验步骤：

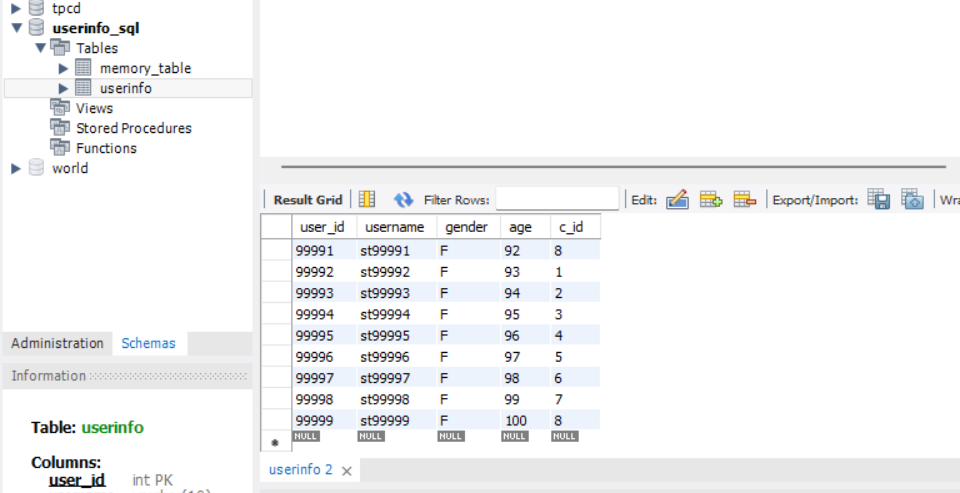
c语言创建txt文件包含数据10万条：



添加数据：



添加成功



实验1：验证有索引和无索引的查询效率差异

首先，你可以执行两个查询，一个使用索引，一个不使用索引，并比较它们的性能。

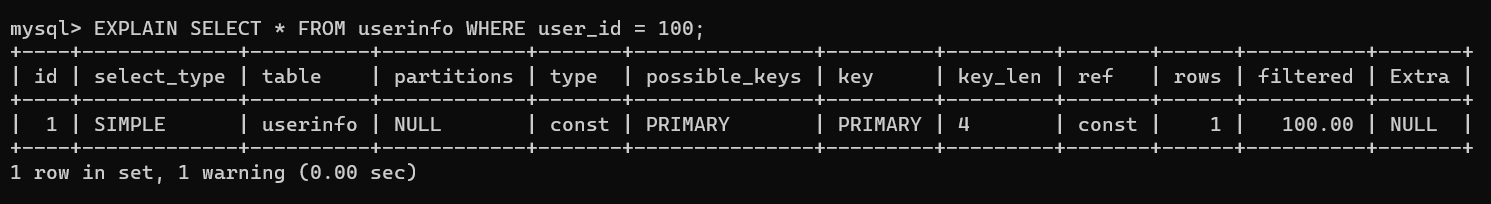
-- 查询使用索引

EXPLAIN SELECT \* FROM userinfo WHERE user\_id = 100;

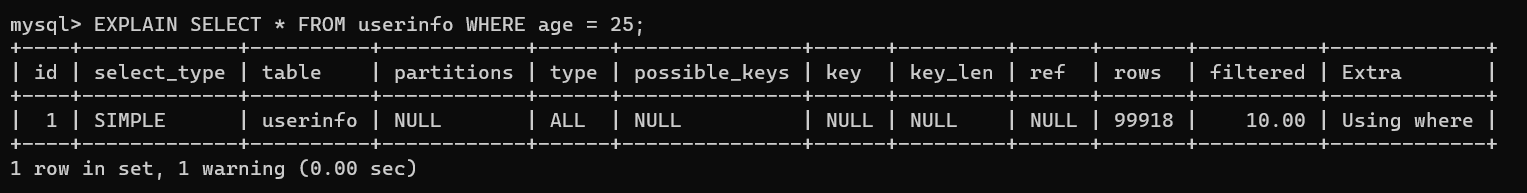
-- 查询不使用索引

EXPLAIN SELECT \* FROM userinfo WHERE age = 25;

使用索引查询：



不使用索引查询：



这个查询没有使用索引，因为age字段没有单独的索引。全表扫描会导致性能较差，特别是在数据量大的情况下。

综合来看，第一个查询因为使用了主键索引而性能较好，而第二个查询由于没有可用的索引而性能较差。这强调了索引的重要性，特别是在大型数据库中，以提高查询性能。

实验2：验证单字段窄索引和多字段构成的宽索引的查询效率

-- 创建单字段索引

CREATE INDEX idx\_user\_id ON userinfo (user\_id);

-- 创建多字段索引

CREATE INDEX idx\_age\_gender ON userinfo (age, gender);

-- 查询使用单字段索引

EXPLAIN SELECT \* FROM userinfo WHERE user\_id = 100;

-- 查询使用多字段索引

EXPLAIN SELECT \* FROM userinfo WHERE age = 25 AND gender = 'F';



查询使用了多字段索引idx\_user\_id 和 idx\_age\_gender，并且表现良好。通常情况下，多字段索引适用于包含所有查询条件的情况，而最左匹配原则意味着索引的最左前缀必须包括在查询中。

这两个查询都使用了索引，并且性能较好。单字段索引适用于单字段的精确匹配查询，而多字段索引适用于包含所有查询条件的情况。

实验3：验证聚集索引与二级索引的查询效率差异（选做）

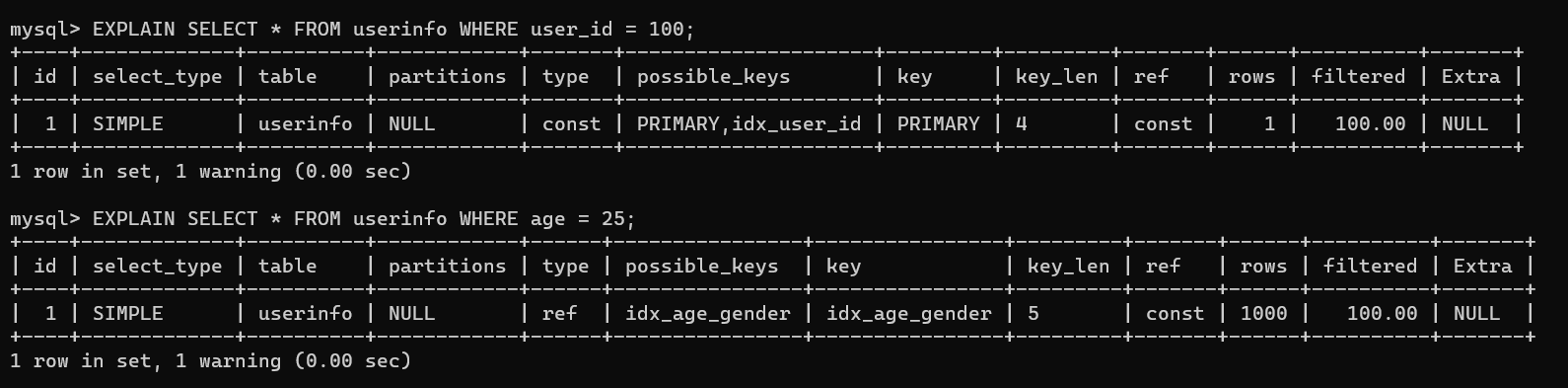
创建聚集索引（主键索引）此处使用userinfo表的主键索引，通常默认是聚集索引。如果需要创建非聚集索引，使用CREATE INDEX语句。

-- 查询使用聚集索引

EXPLAIN SELECT \* FROM userinfo WHERE user\_id = 100;

-- 查询使用非聚集索引

EXPLAIN SELECT \* FROM userinfo WHERE age = 25;



查询1：EXPLAIN SELECT \* FROM userinfo WHERE user\_id = 100;

在这个查询中，你使用了主键索引（聚集索引），因为你在WHERE子句中使用了主键字段user\_id。

分析：这个查询使用了主键索引，它是最有效的索引类型，因为它在唯一性上具有最好的性能。这是一个非常快速的点查询，只匹配了1行。

查询2：EXPLAIN SELECT \* FROM userinfo WHERE age = 25;

在这个查询中，你使用了多字段索引 "idx\_age\_gender"，因为你在WHERE子句中使用了 "age" 和 "gender" 字段。

分析：这个查询使用了多字段索引 "idx\_age\_gender"，它允许范围查询。虽然它不如主键索引那么高效，但对于范围查询来说仍然是有效的。在这种情况下，它可能匹配了1000行。

总的来说，这两个查询都是有效的，但第一个查询使用主键索引，更快速，而第二个查询使用多字段索引，在范围查询时也表现良好。通常情况下聚集索引性能更好，但具体结果可能取决于查询的性质和数据库引擎。

实验4：验证B树索引与Hash索引的查询效率差异（选做）

验证B树索引和Hash索引的性能差异，可以使用MySQL的Memory引擎。

-- 创建内存引擎表

CREATE TABLE memory\_table

(

user\_id INT,

username VARCHAR(10),

age INT,

INDEX idx\_btree\_age (age) USING BTREE,

INDEX idx\_hash\_age (age) USING HASH

) ENGINE = MEMORY;

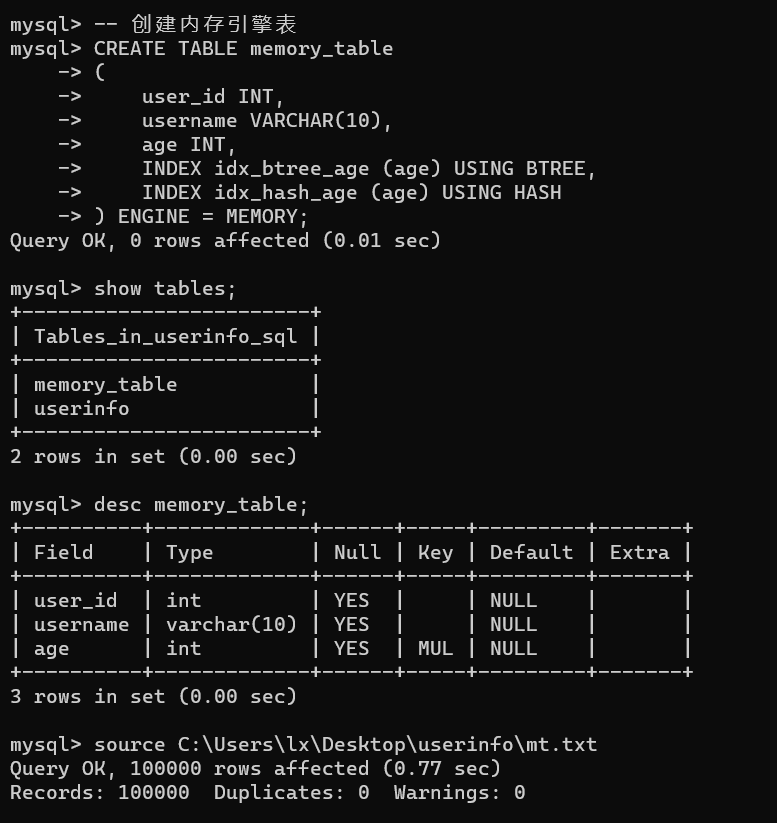
插入数据到内存表

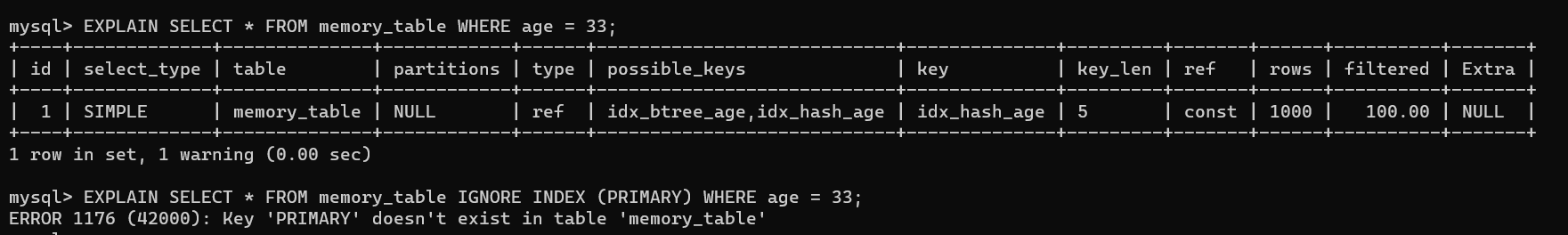
-- 查询使用B树索引

EXPLAIN SELECT \* FROM memory\_table WHERE age = 33;

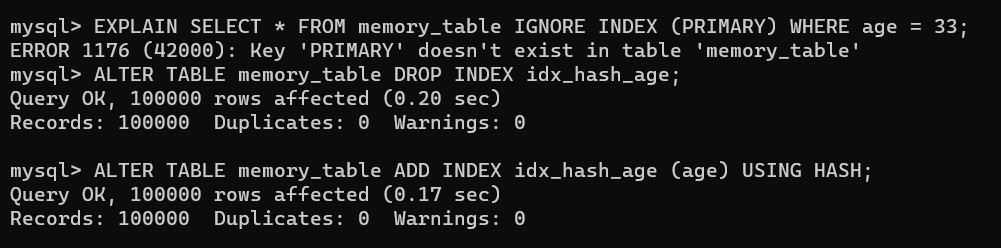
-- 查询使用Hash索引

EXPLAIN SELECT \* FROM memory\_table IGNORE INDEX (PRIMARY) WHERE age = 33;

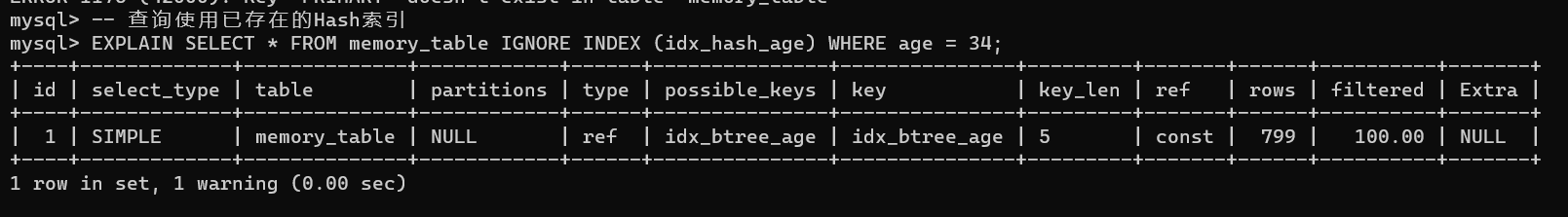




更新索引：



查询相关HASH：



总结：

对于完全匹配的点查询，Hash索引和B树索引都可以很好地工作，而且在你的示例中，它们都选择了正确的索引。

两个查询都有相似的性能，但在实际情况下，性能可能会取决于索引的大小、数据分布等因素。

这两个查询的性能差异可能不明显，因为你的数据集较小。通常，Hash索引在点查询时会非常快，而B树索引在范围查询时更有效。性能差异可能会在大型数据集上更加显著。

**题目3 .**参考《GaussDB(for MySQL)数据库维护管理指导手册.docx》，在华为云数据库上的DAS端了解以下功能。（选做实验）

(1)1.2小节管理数据库集群。

特别注意：该小节实验仅做查看操作！增加集群节点，集群规格变更，重置管理员账户名密码，集群备份恢复，参数修改等对全局有影响的变更操作仅做了解，不要实际操作！

(2)1.3.1性能监控。

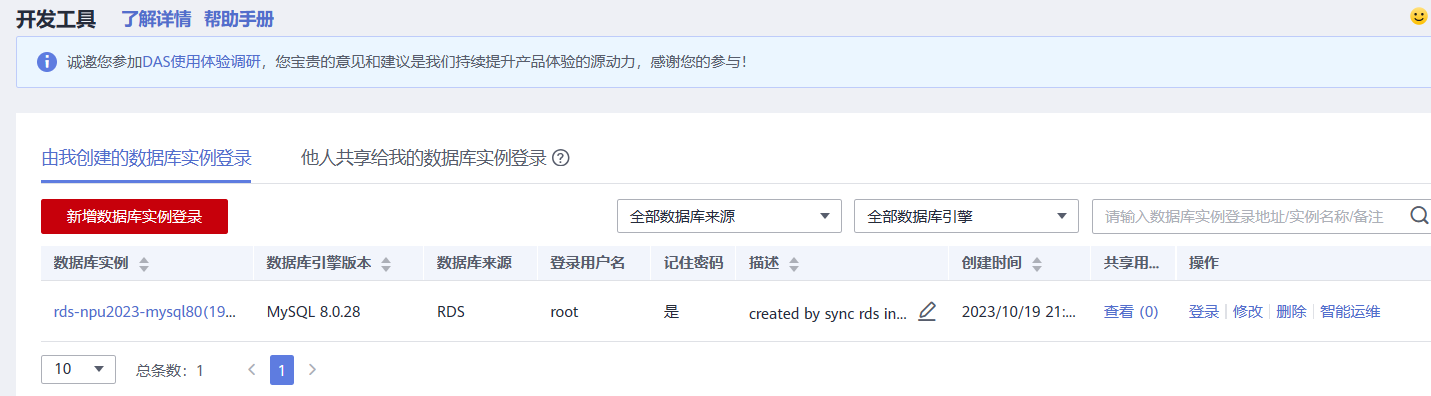
(3)1.3.2 性能调优。

实验过程：

1.2

查看数据库实例

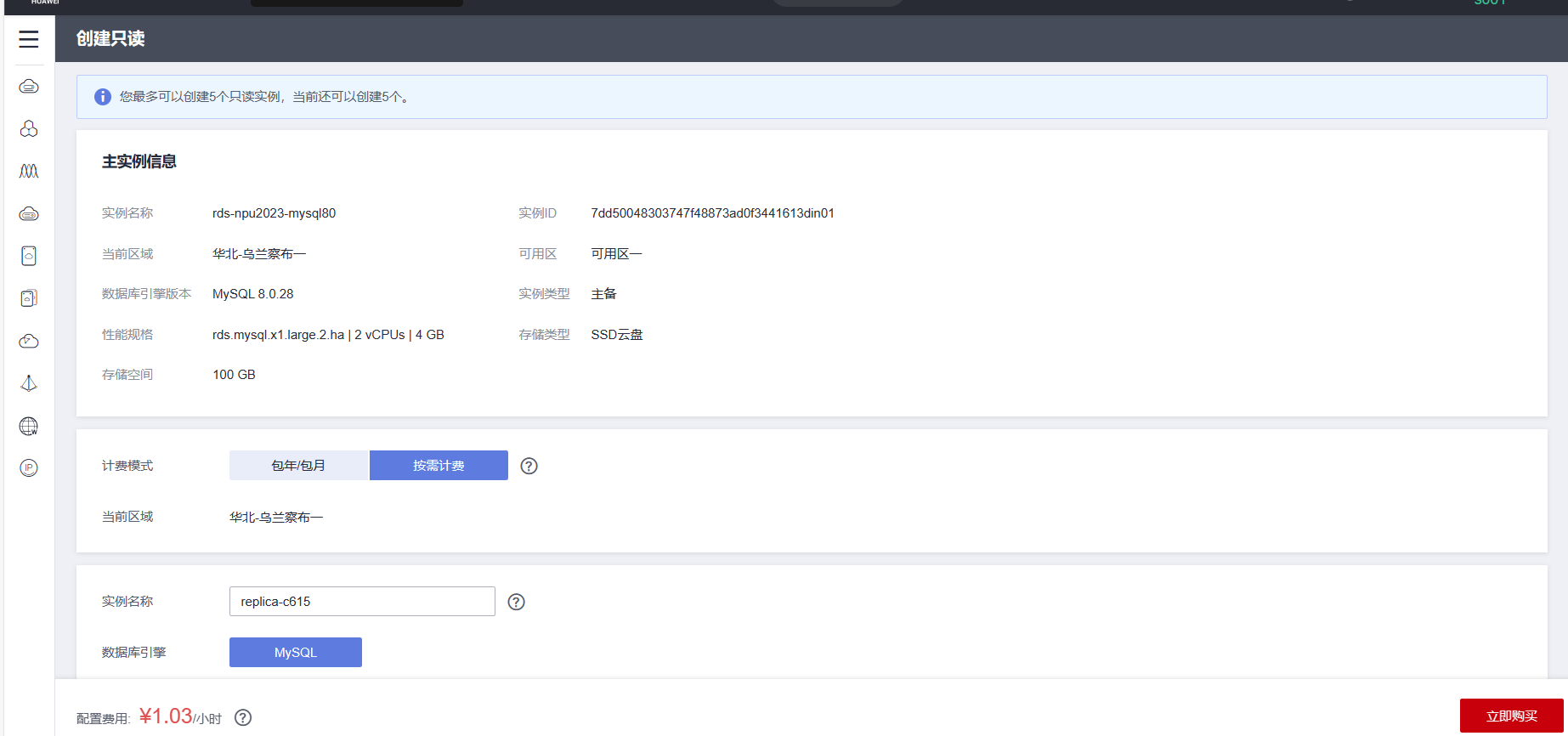
在“数据管理服务 DAS”页面中，选择需要查看的数据库实例





增加集群节点

在“基本信息-实例拓扑图”界面，点击“创建只读”。

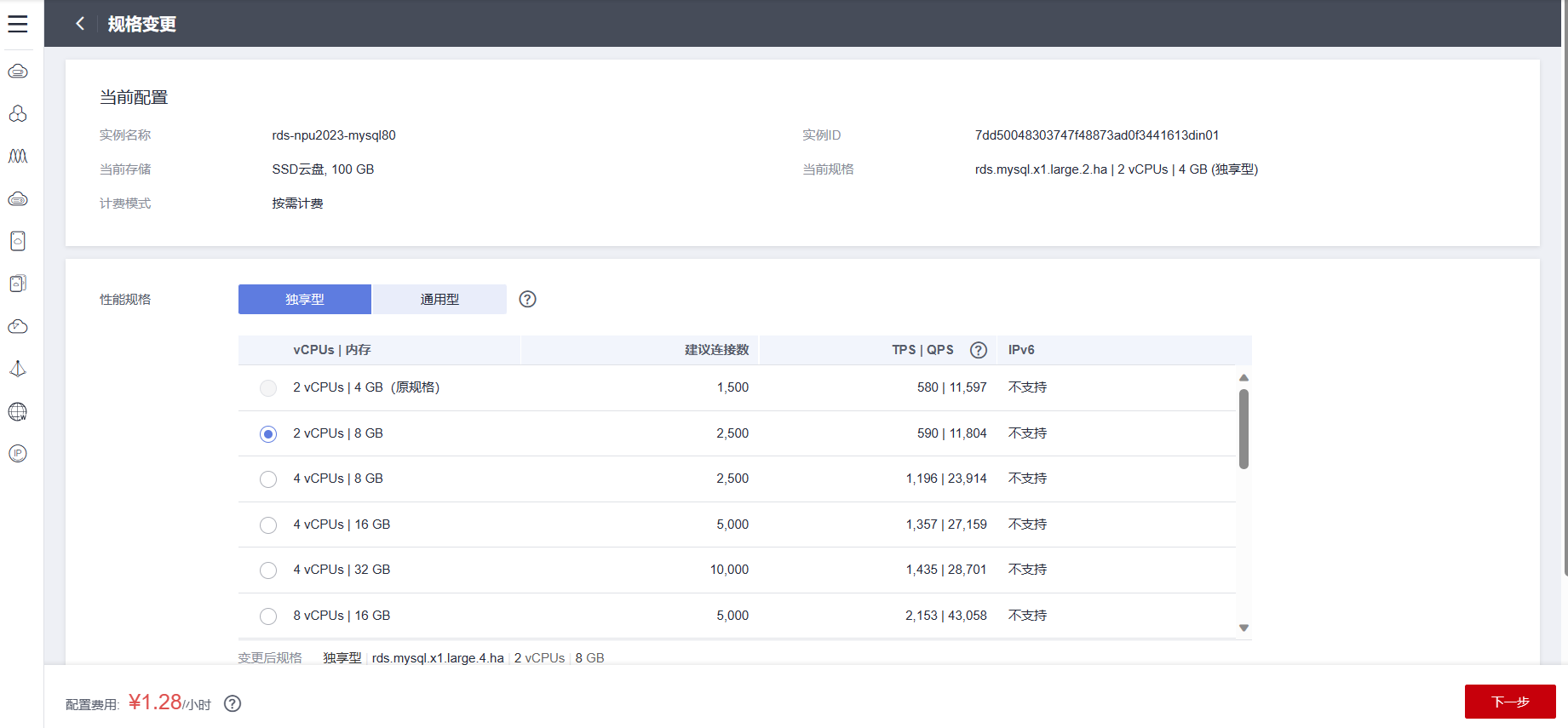


查看信息即可，不用进行购买增加操作。

集群规格变更

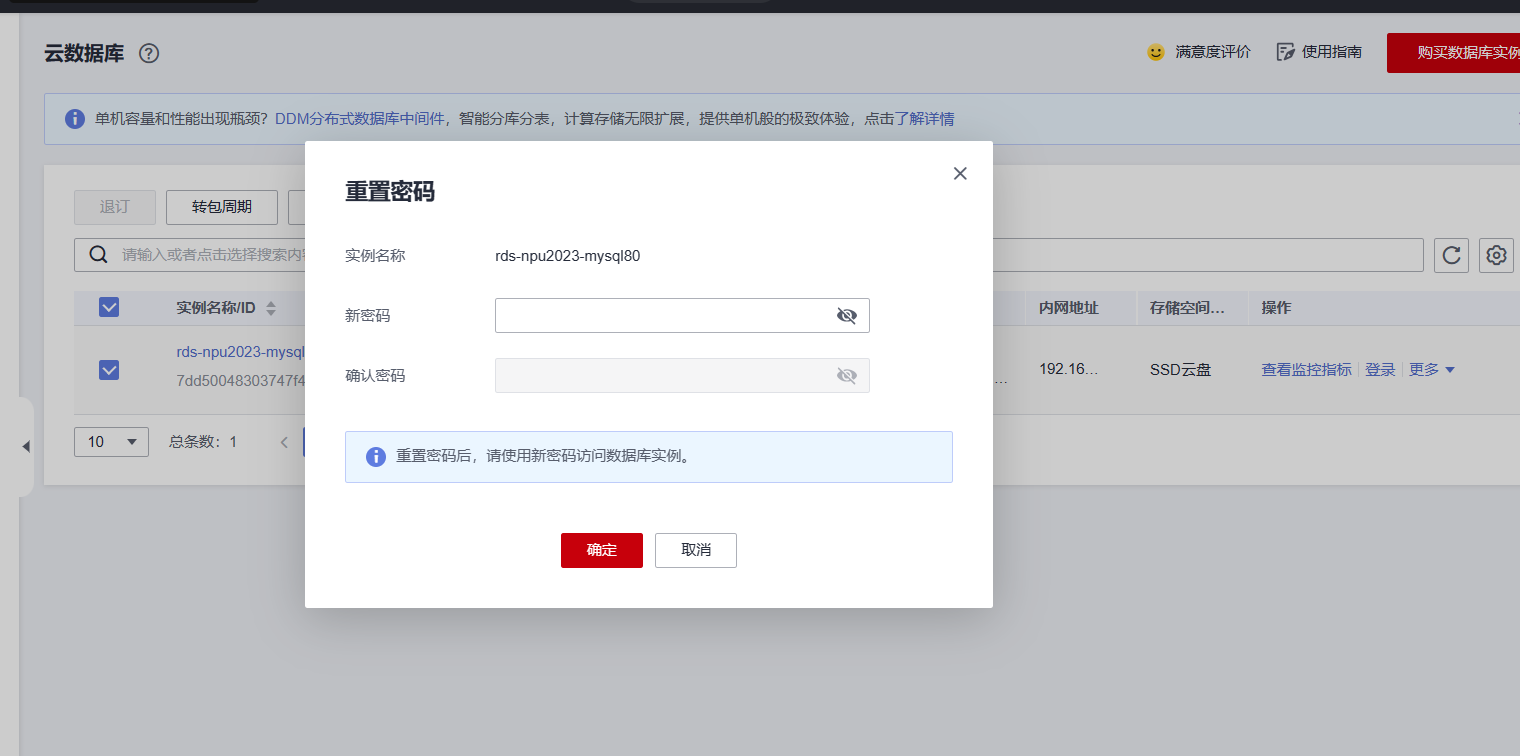
说明：此处集群规格变更只为了说明扩容方法，请根据实际业务情况进行选择。

在“基本信息-实例信息”界面，“性能规格”一栏点击“规格变更”。



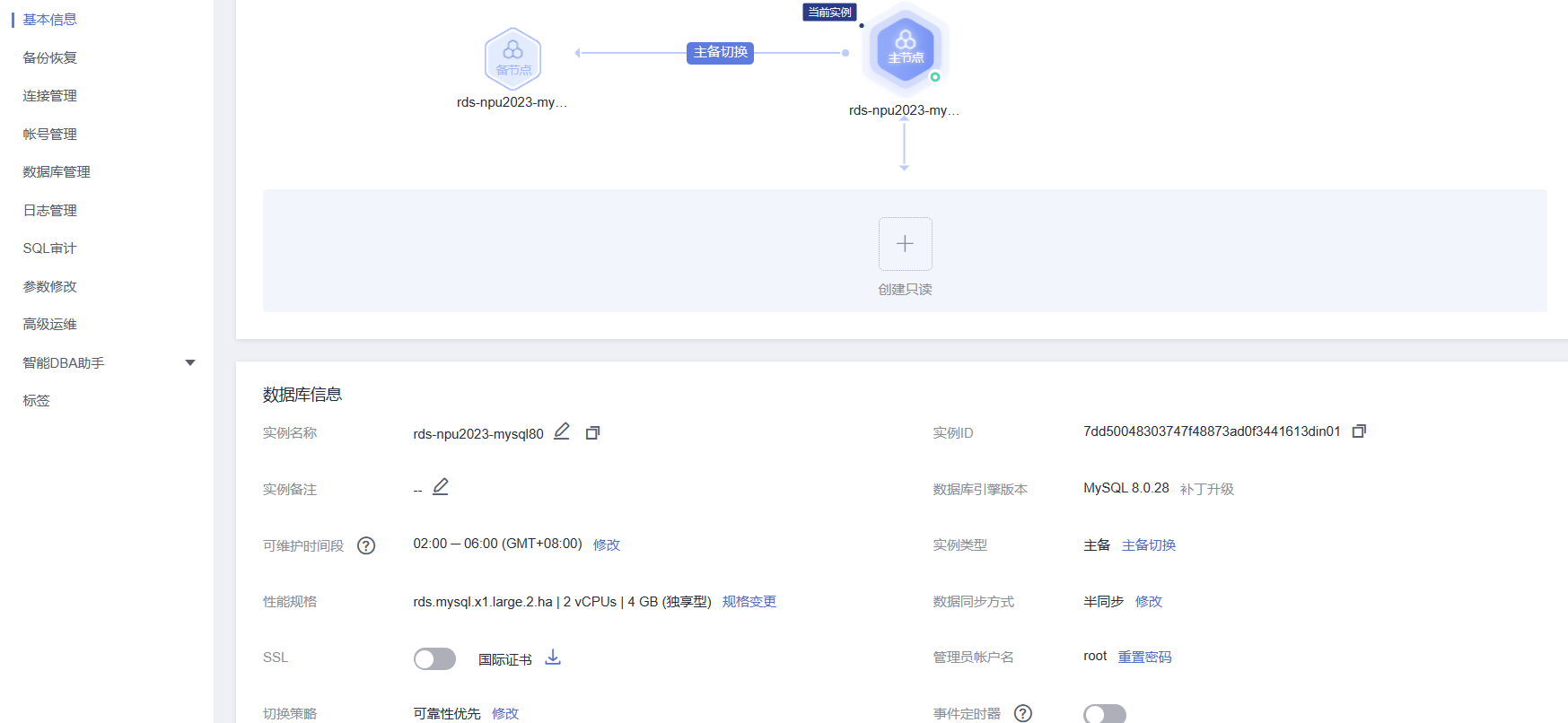
在此查看规格信息，也不需要更改集群规格。

重置密码界面如下：



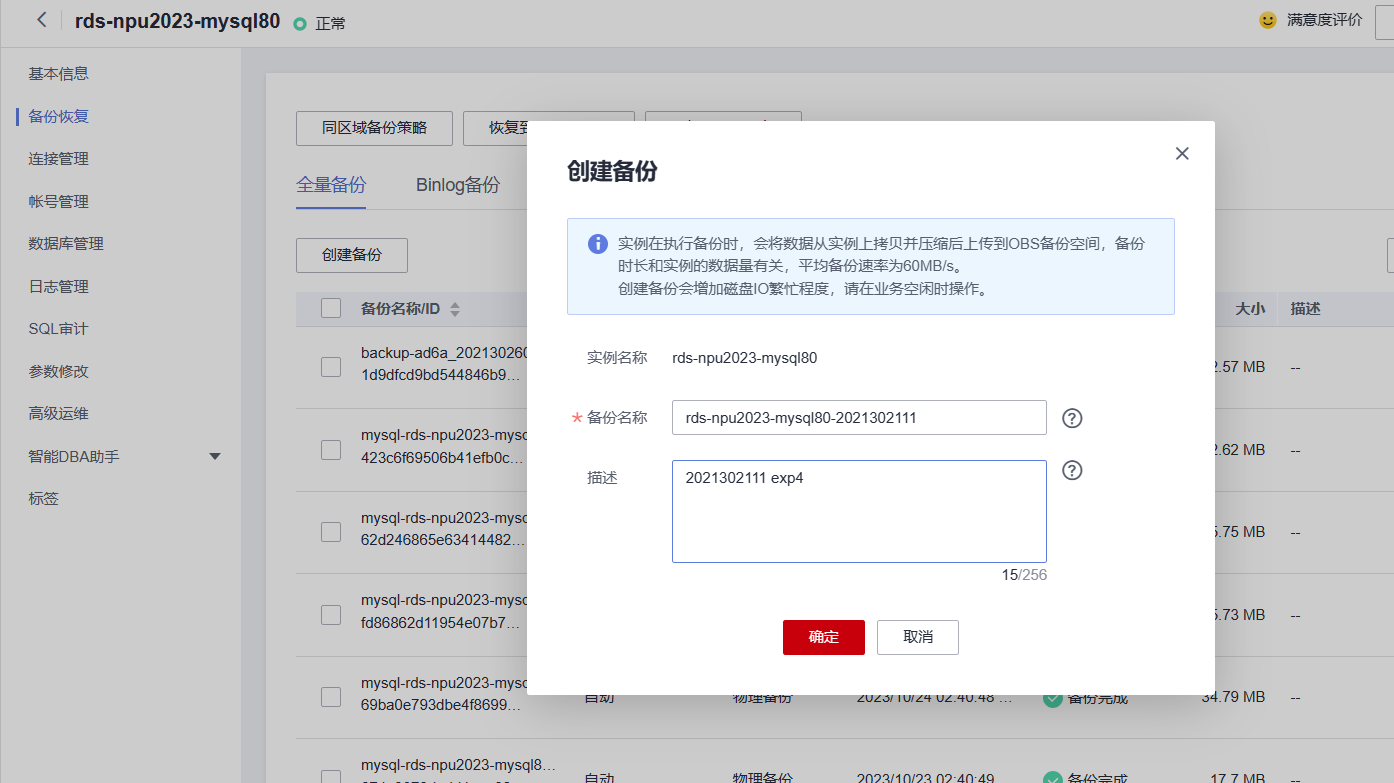
查看集群网络信息

网络信息如下：





创建备份恢复：



也可以进行修改备份策略，恢复制定时间点等步骤：



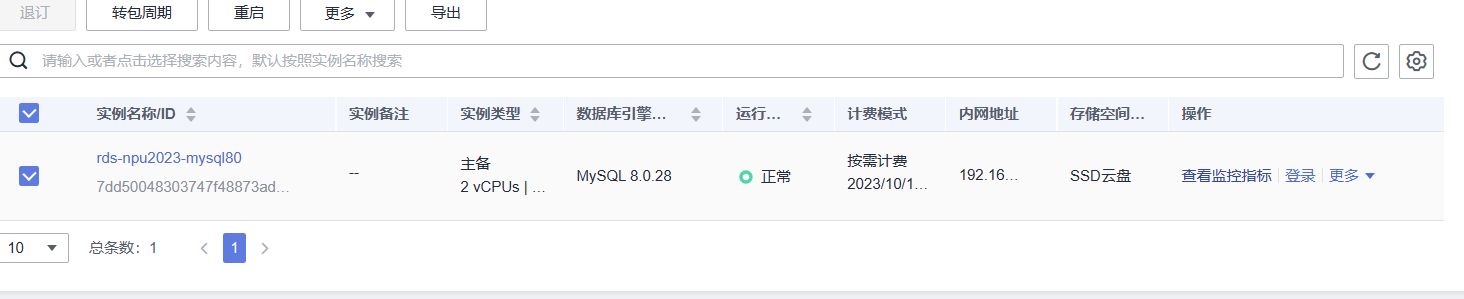
日志管理处可以查询各种类型的日志并进行参数修改：



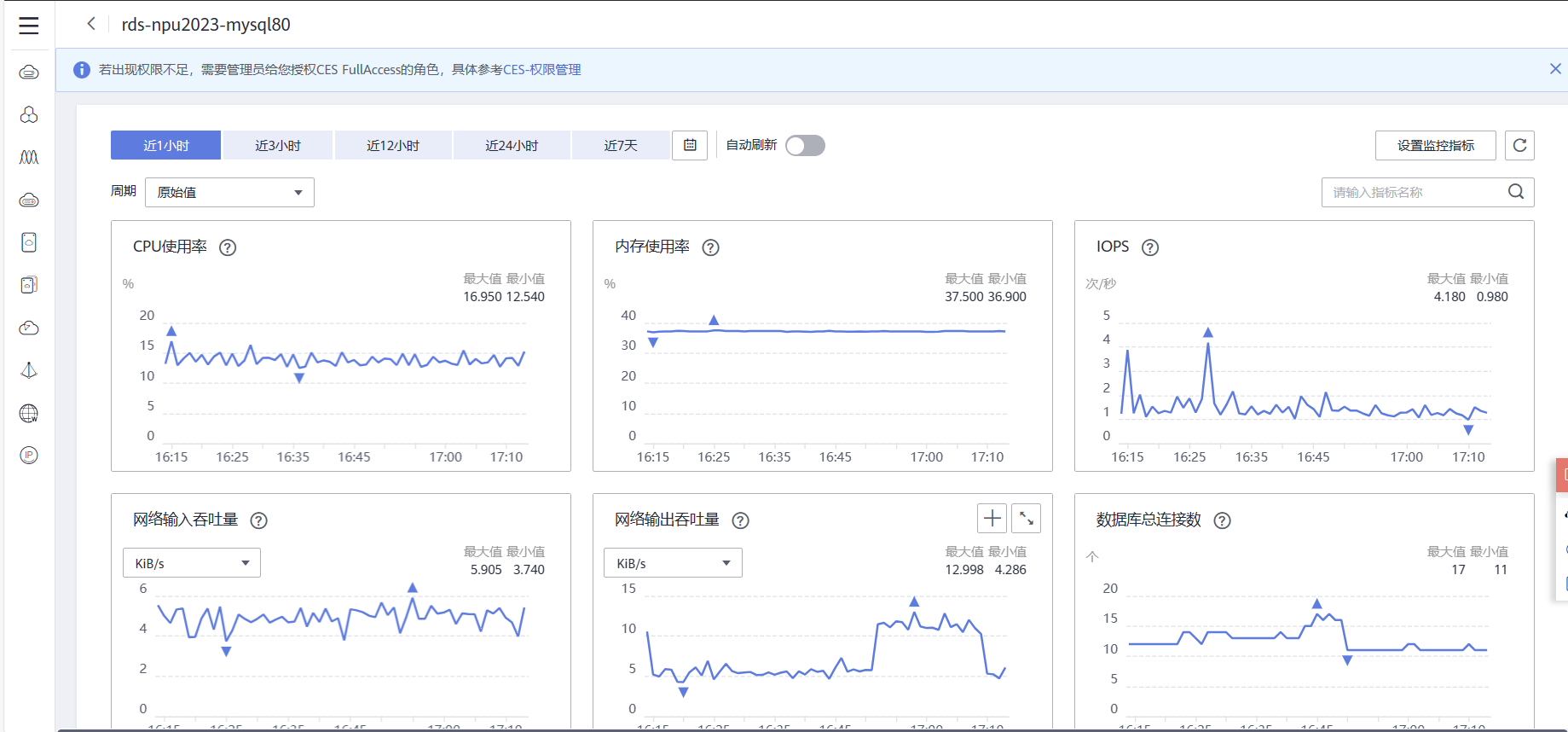
1.3.1

1.3.1 性能监控

步骤 1 在“基本信息-节点信息”页面，在节点操作栏中点击“查看监控”。

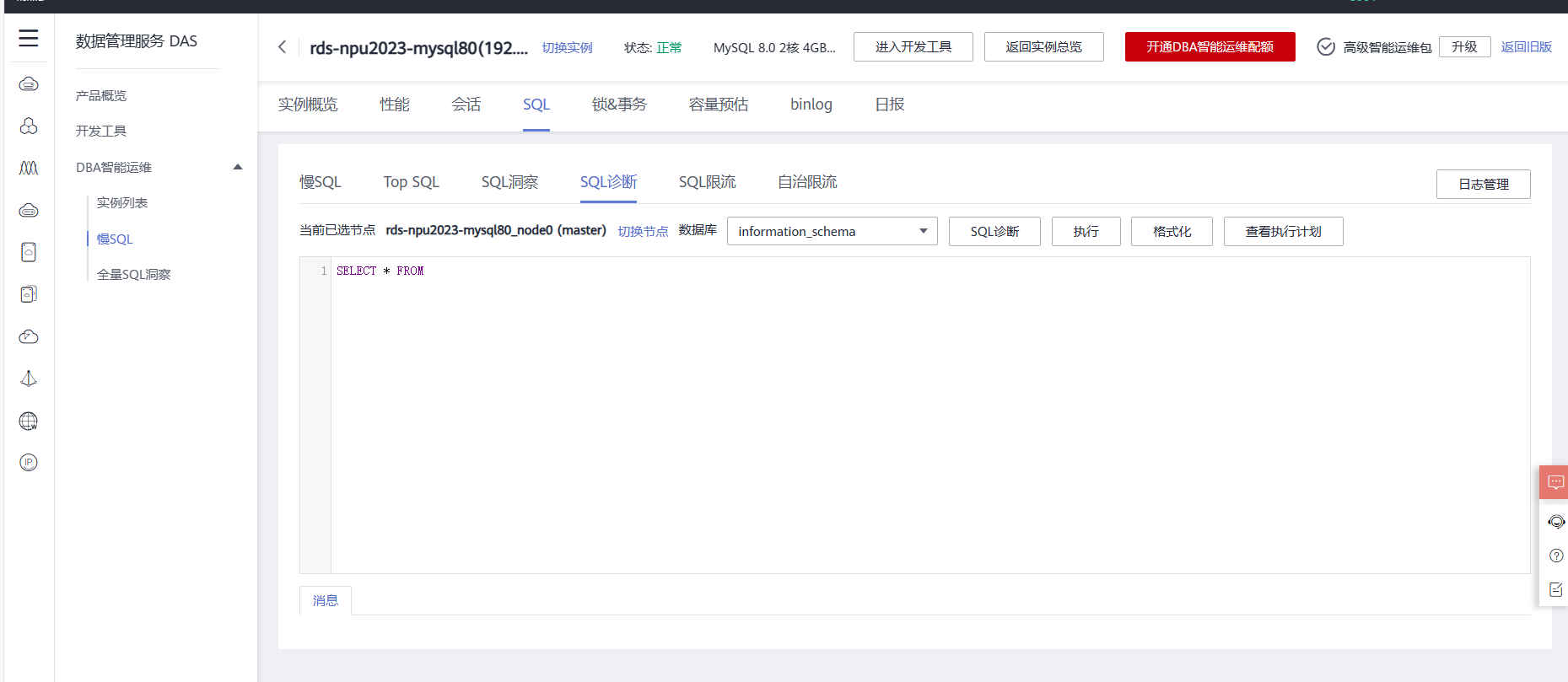


步骤 2 查看性能监控。

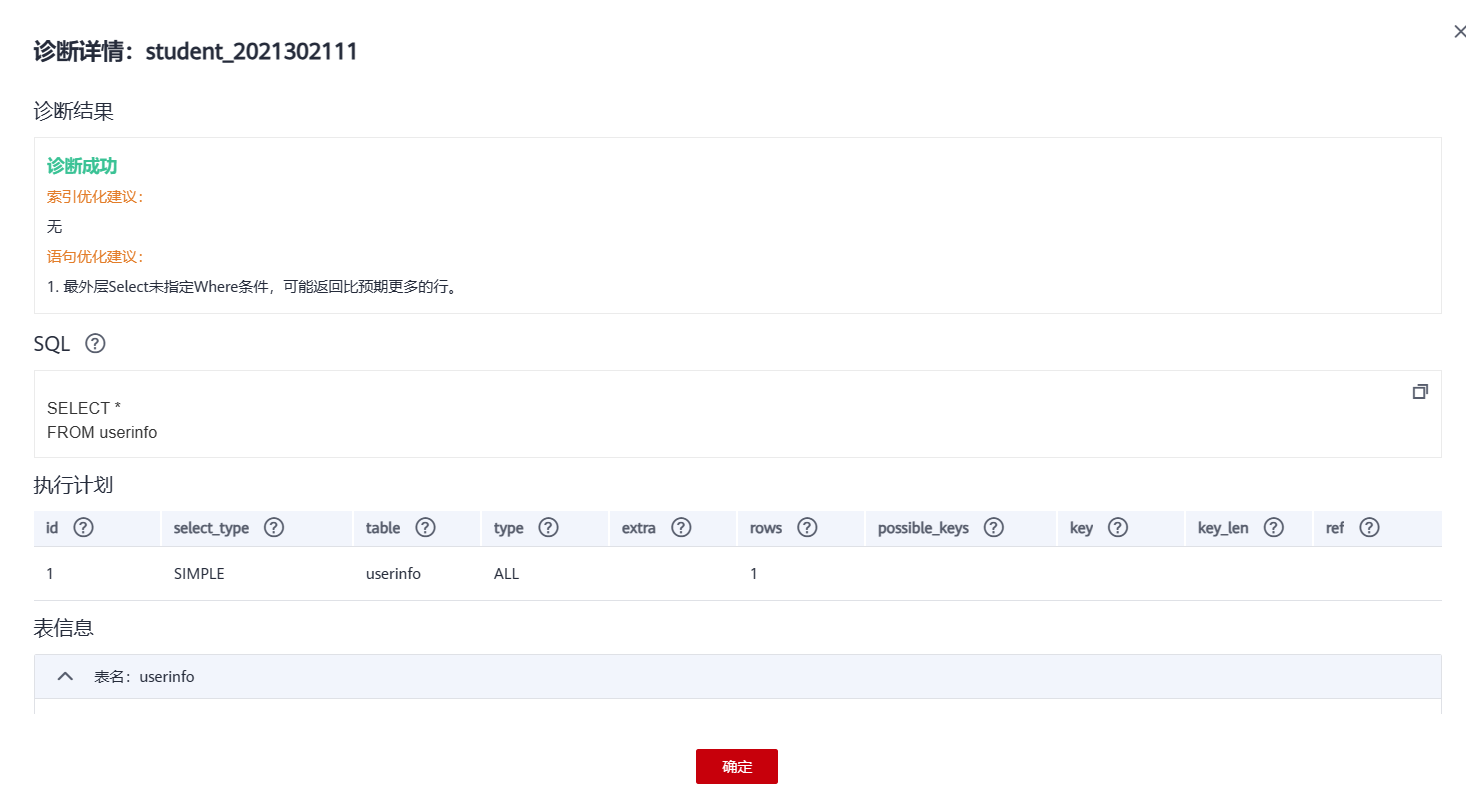


1.3.2 性能调优

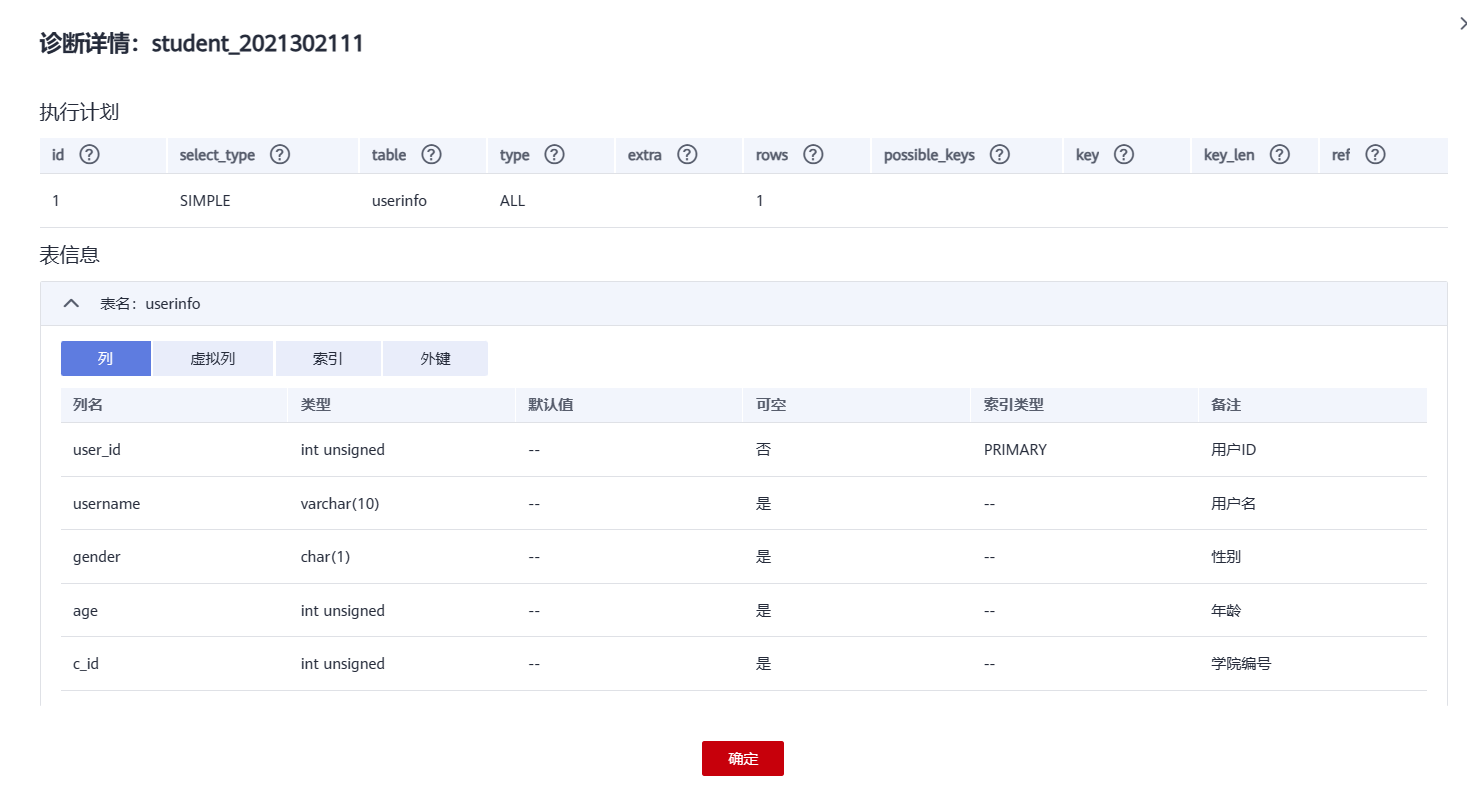
步骤 1 通过 DAS 登录数据库，在“智能运维”下拉菜单中选择“SQL 诊断”。



步骤 2 在“SQL 诊断”页面中，点击“新增 SQL 性能诊断”。



输入语句进行诊断，得出诊断结果和表信息，至此实验结束。



**三 .实验中出现的问题以及解决方案**

**问题及解决：此次实验的难点在于explain结果的分析反思和业界华为云的使用方法与了解，通过老师提供的资料和讲解，完成了任务。**

**感受与建议：学习SQL的视图和索引是数据库管理和查询优化的重要方面。视图允许将查询结果封装成虚拟表，简化了复杂查询和数据保护，提高了数据库的安全性和可维护性。索引则是提高查询性能的关键，它加速了数据检索，减少了查询时间。通过学习视图和索引，我能更高效地管理数据和提高查询性能，从而提供更好的数据库体验，不仅在日常工作中有用，也有助于数据库设计和性能优化的深入理解。**

**批阅者：**

**批阅日期：**

**实验成绩：**

**批注：**