### 实验2 数据库的创建和管理

**实验学时：1学时**

**实验类型：验证**

**实验要求：必做**

**一、实验目的**

熟练掌握使用Navicat和Transact-SQL语言两种方法创建、修改和删除数据库。学习在Navicat中进行数据库的转储和导入。掌握管理数据库的有关系统存储过程。

**二、实验内容**

1、利用Navicat创建满足以下要求的数据库：

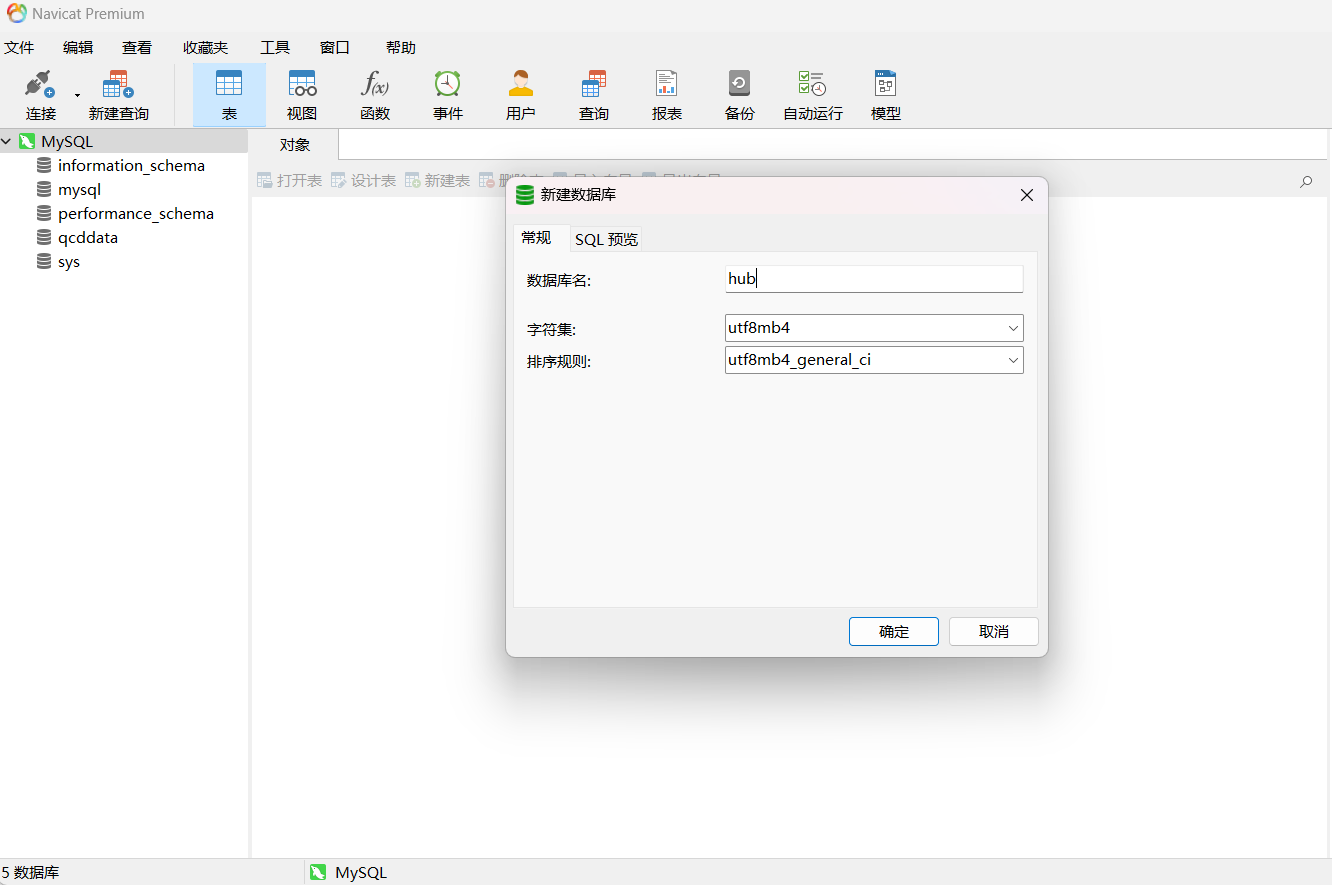
1）数据库存在于连接MySQL中；

2）数据库名称为hub；

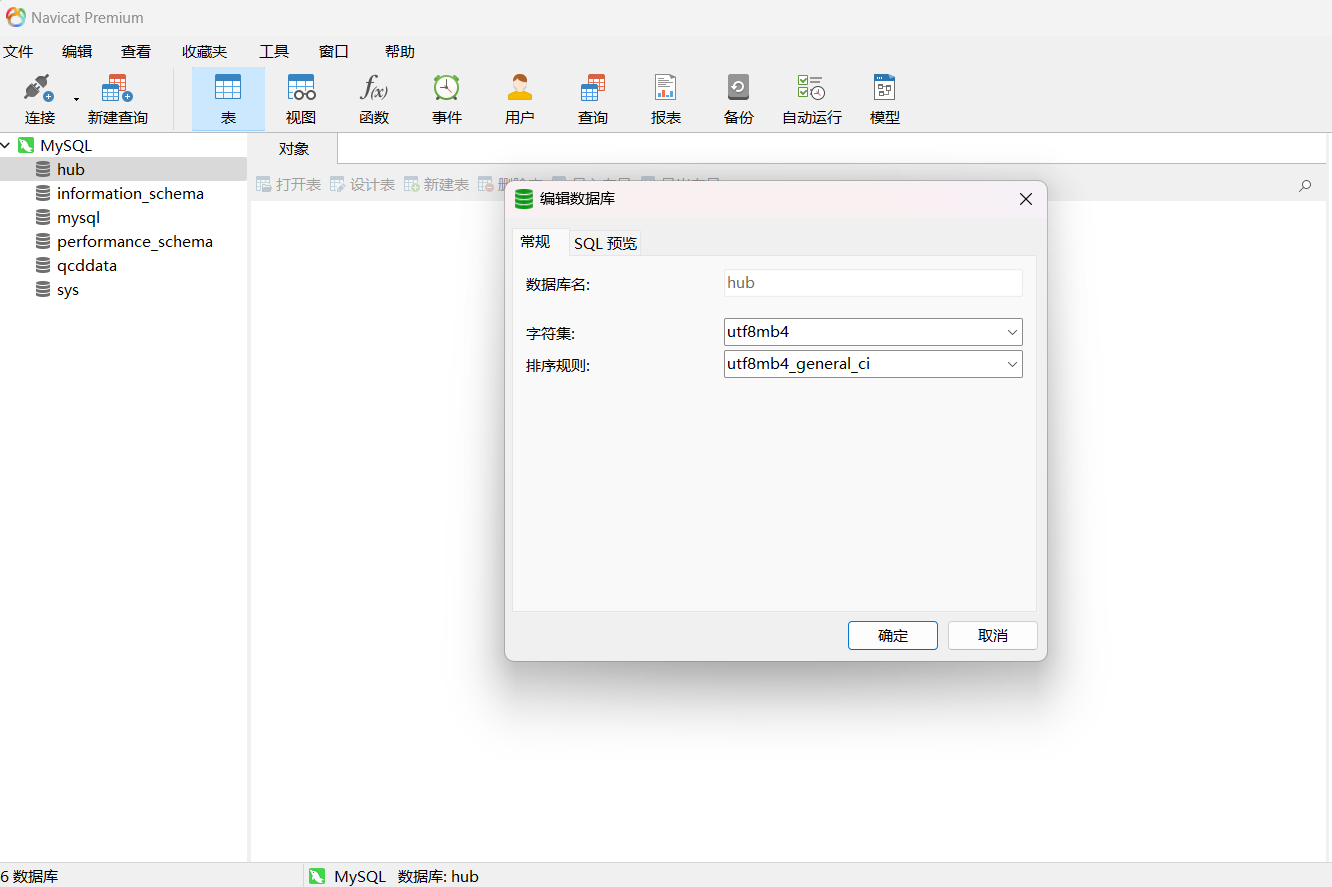
3）字符集选择utf8mb4；

4）排序规则选择utf8mb4\_general\_ci

具体步骤如下：打开“MySQL”连接—单击选择并右键—“新建数据库”—输入数据库名“mydb”、字符集“utf8mb4”、排序规则“utf8mb4\_general\_ci”—“确定”。如图所示：



右键—编辑数据库，可以看到如图所示结果：



2、利用Transact-SQL语言创建满足以下要求的数据库：

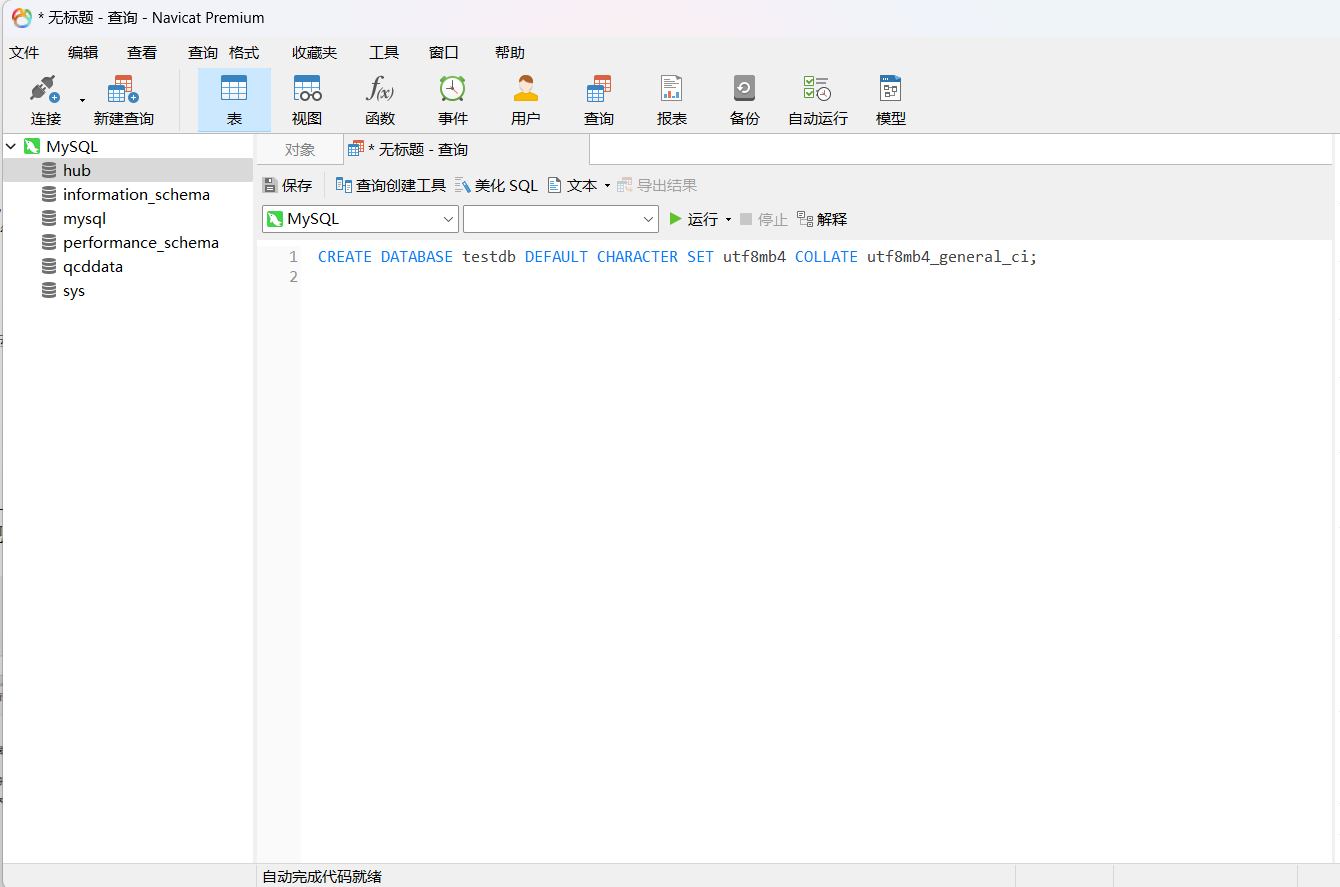
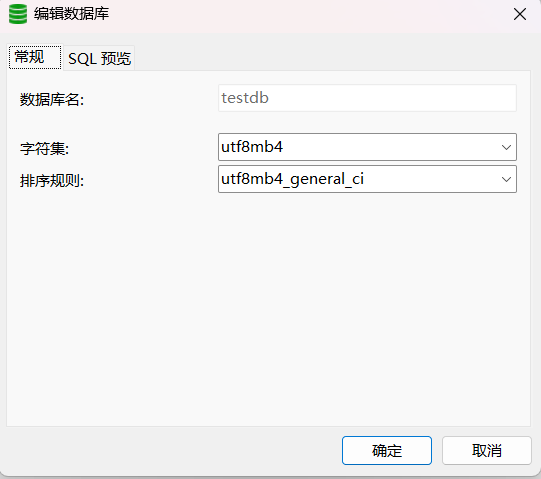
1）数据库存在于连接MySQL中；

2）数据库名称为testdb；

3）字符集选择utf8mb4；

4）排序规则选择utf8mb4\_general\_ci；

具体步骤如下： 点击“新建查询”，在查询编辑器输入以下代码，点击“运行”，完成后，左侧列表中选中连接“数MySQL”，点击“刷新”按钮，列表即可见到“testdb”。如图所示：

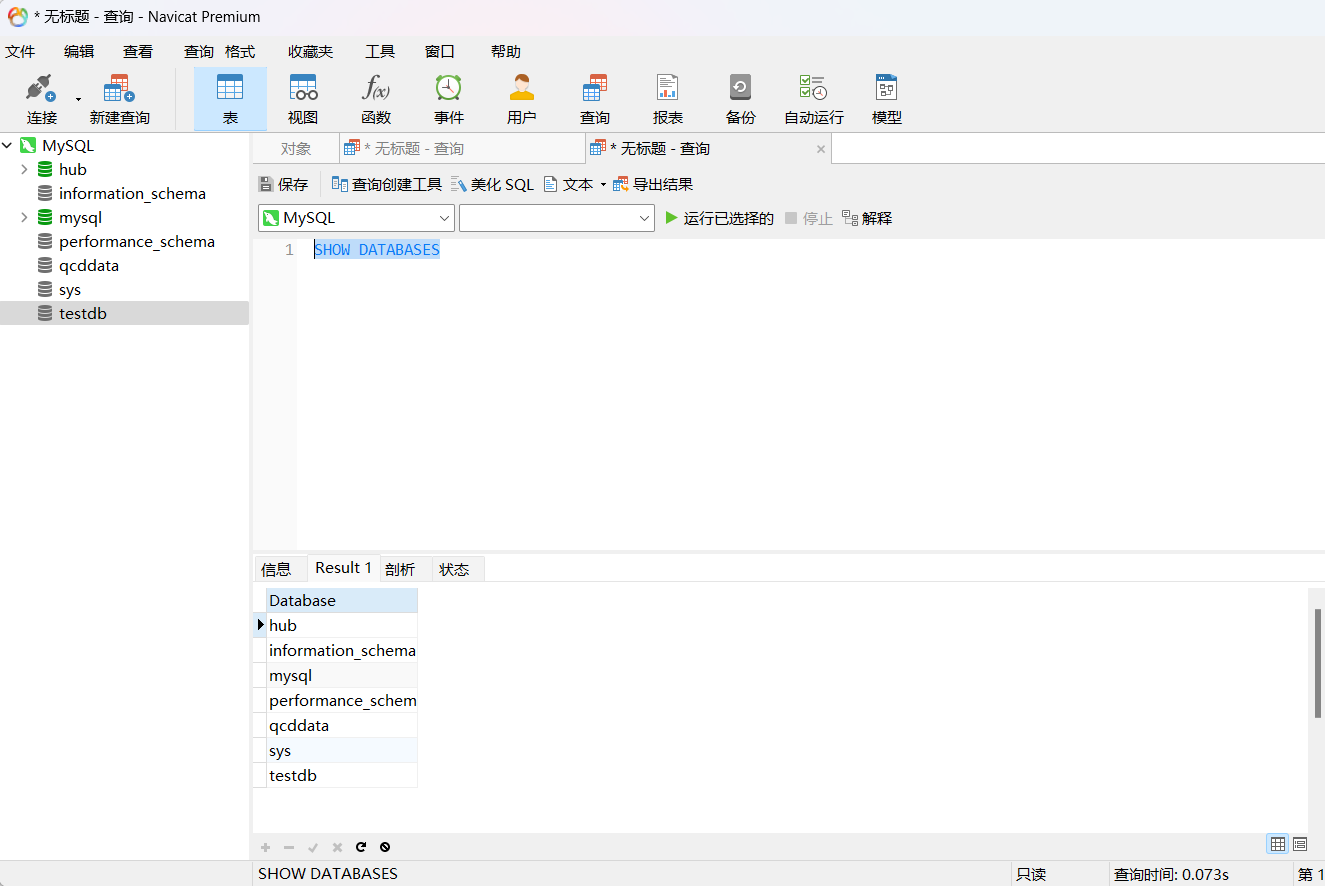
3、利用Transact-SQL语言查看数据库及表的信息。

①查看连接中的所有数据库

步骤：点击“新建查询”，输入

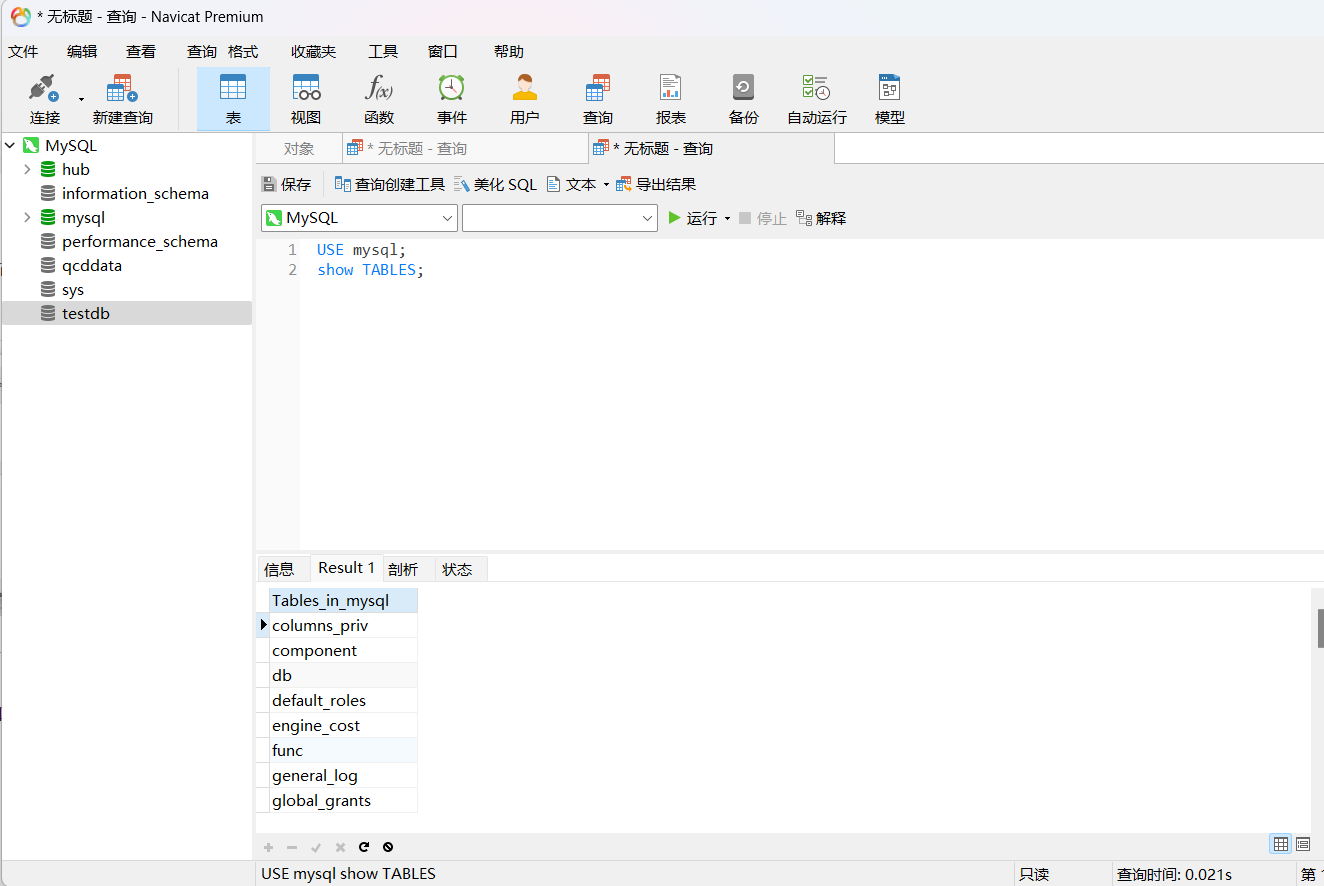
SHOW DATABASES

后运行，结果如图所示：



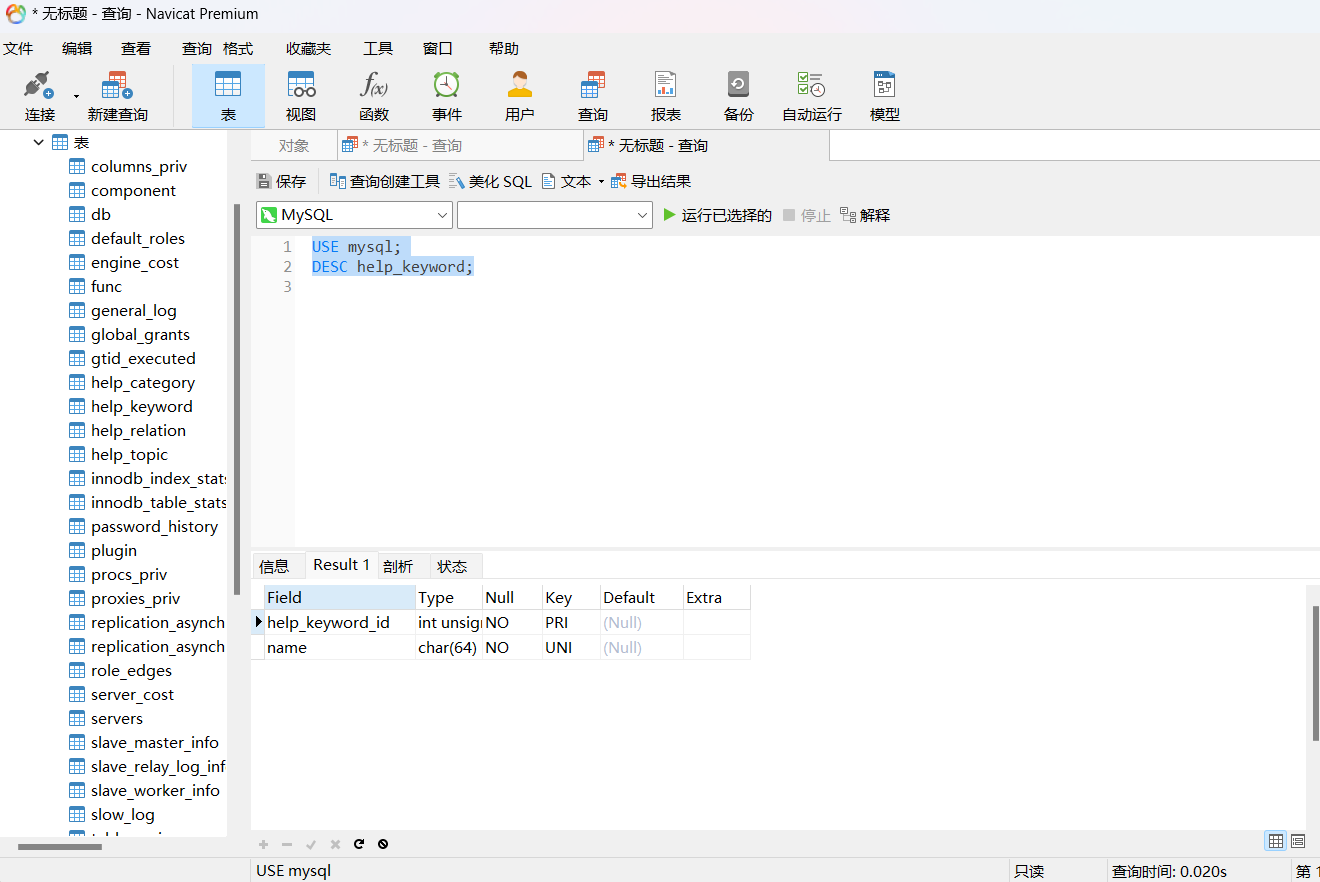
②查看数据库mysql中所有的表

步骤：点击“新建查询”，输入指令后运行，（也可以用语句show tables from mysql;）结果如图所示：



③查看数据库“mysql”中表“help\_keyword”的结构。

步骤：点击“新建查询”，输入后运行，结果如图所示：



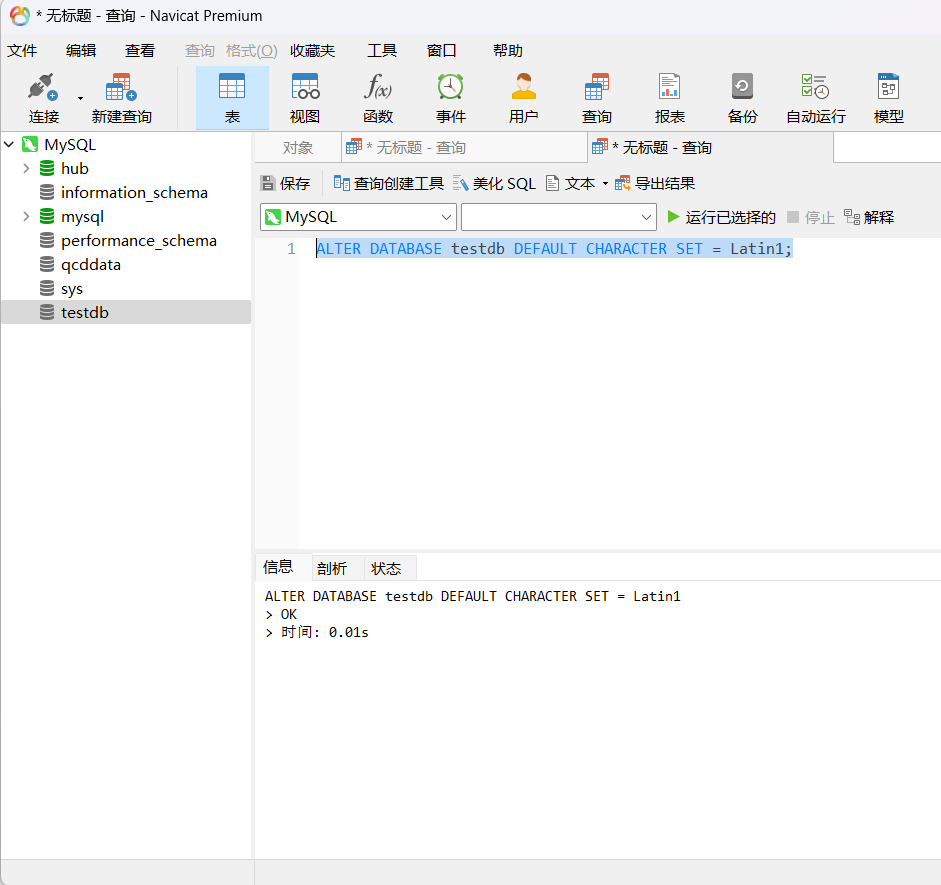
4、利用Navicat修改数据库testdb，修改其排序方式为utf8mb4\_bin。

步骤：右键单击数据库“testdb”—编辑数据库，在排序方式中选择“utf8mb4\_bin”—“确定”。

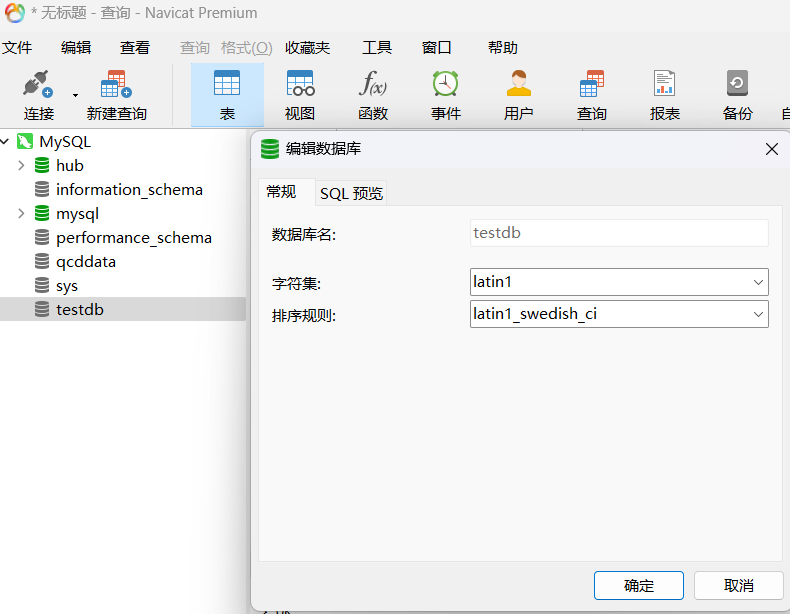
5、利用Transact-SQL语言修改数据库testdb的字符集为Latin1；

步骤：点击“新建查询”，输入以下代码，点击运行

如图所示：



关闭数据库“mydb”后重新打开，查看数据库属性，可以看到如图所示结果：



1. 利用Navicat或利用Transact-SQL删除数据库testdb

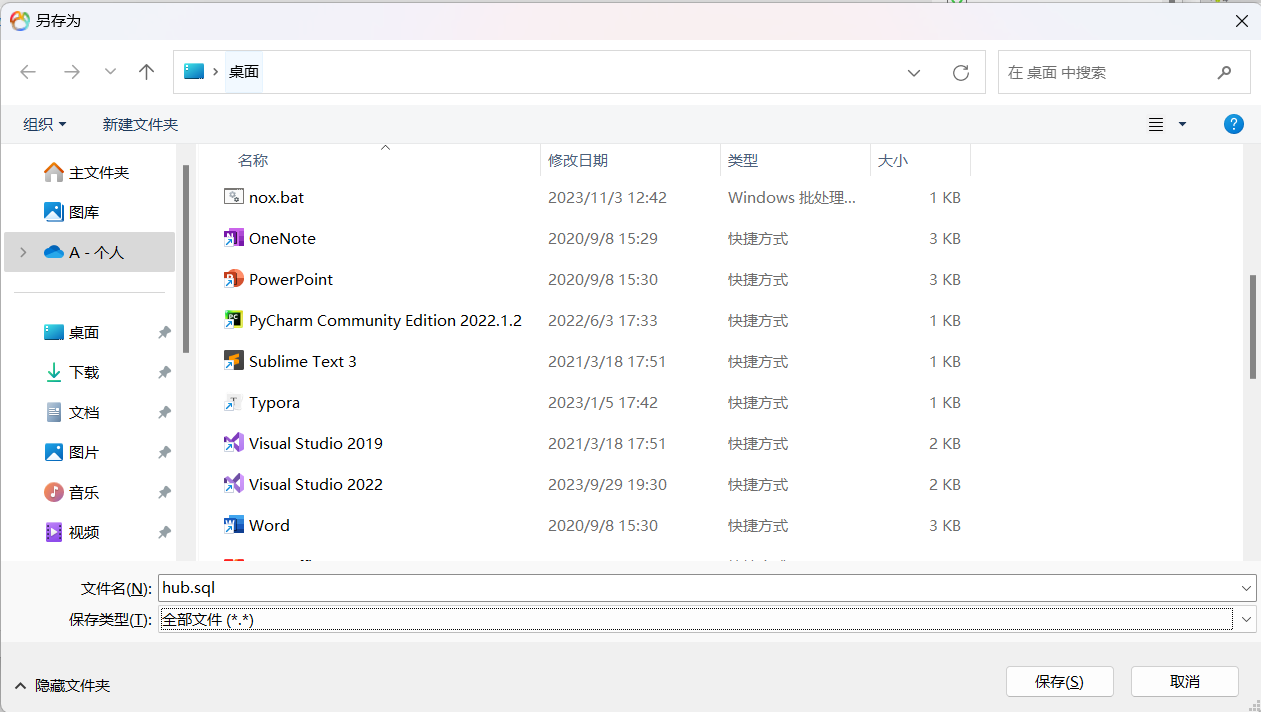
（1）利用Navicat删除数据库：直接右键点击testdb选择删除即可，注意要关闭当前的SQL语句编写界面。

（2）利用Transact-SQL语言删除数据库testdb。

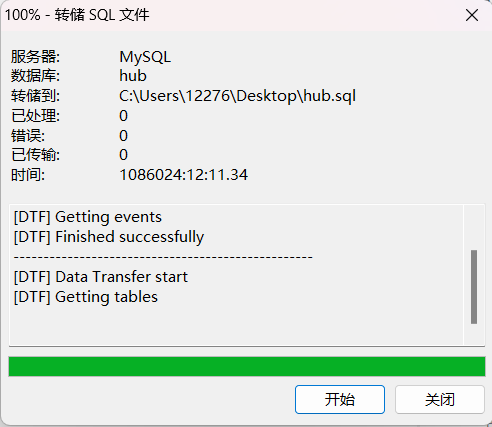
步骤：点击“新建查询”，输入DROP DATABASE testdb 后运行。刷新连接“MySQL”，查看结果。

7、转储MySQL数据库。

步骤：打开连接“MySQL”—右键单击数据库“hub”—选择“转储SQL文件”—将文件名改为“hub.sql”后保存在桌面，如图所示：



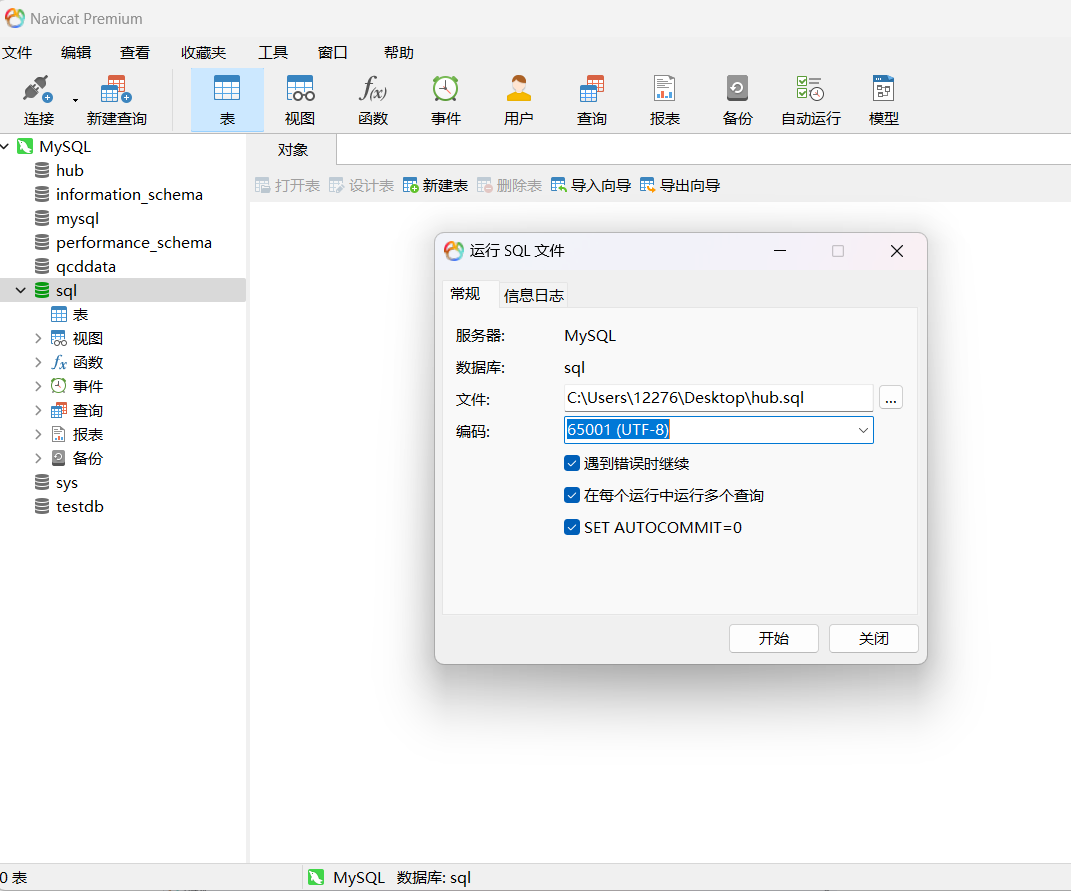
成功后将出现如图所示情况：



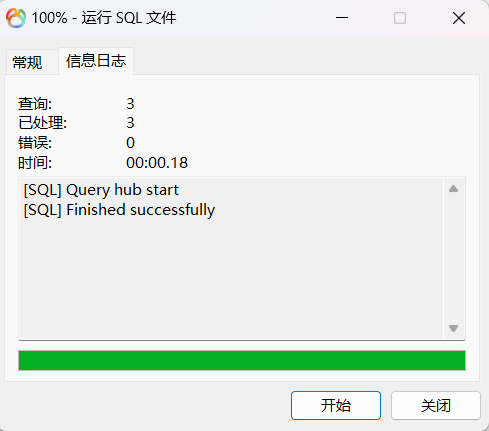
8、导入MySQL数据库。

①在连接“MySQL”中新建数据库“sql”（字符集与排序方式尽量一致）。

②打开数据库“sql”—右键单击—运行SQL文件—选择桌面的“hub.sql”文件—点击“确定”。如图所示：



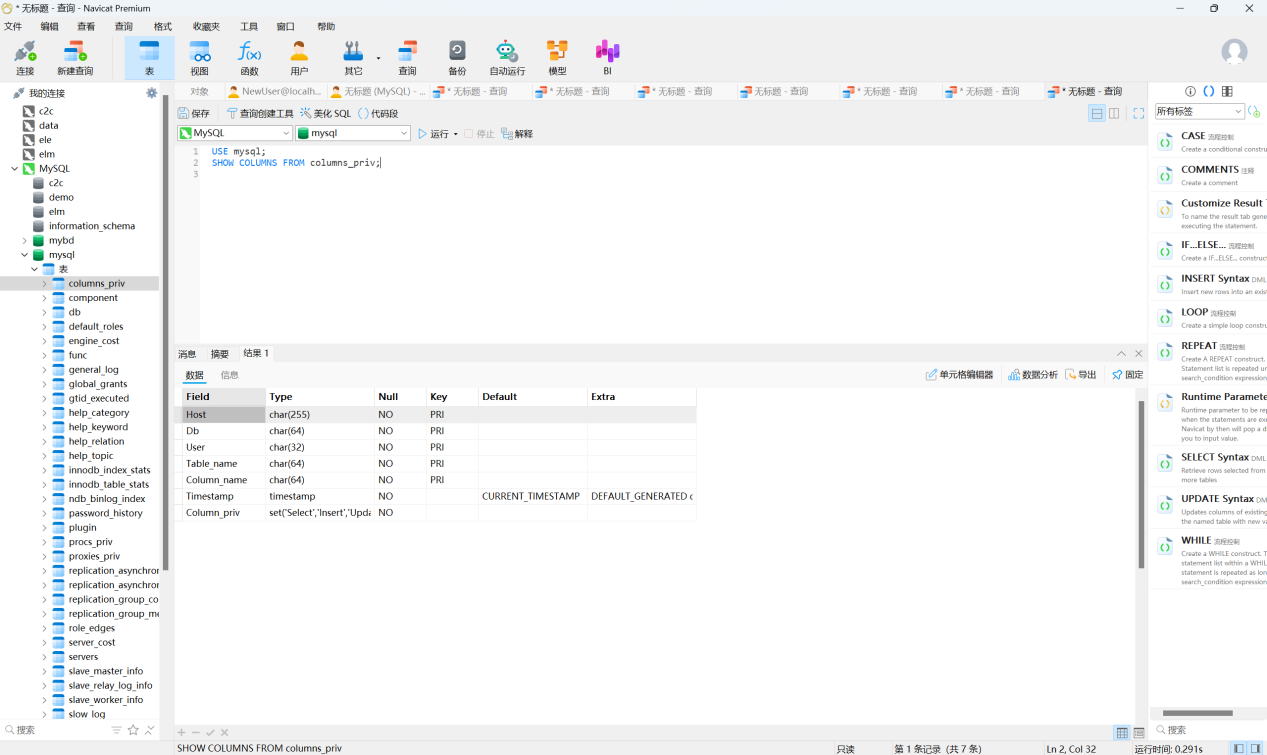
③执行结果如图所示则表示成功。关闭数据库sql后重新打开，对比数据库“sql”与最初的数据库“hub”，看是否完全相同。



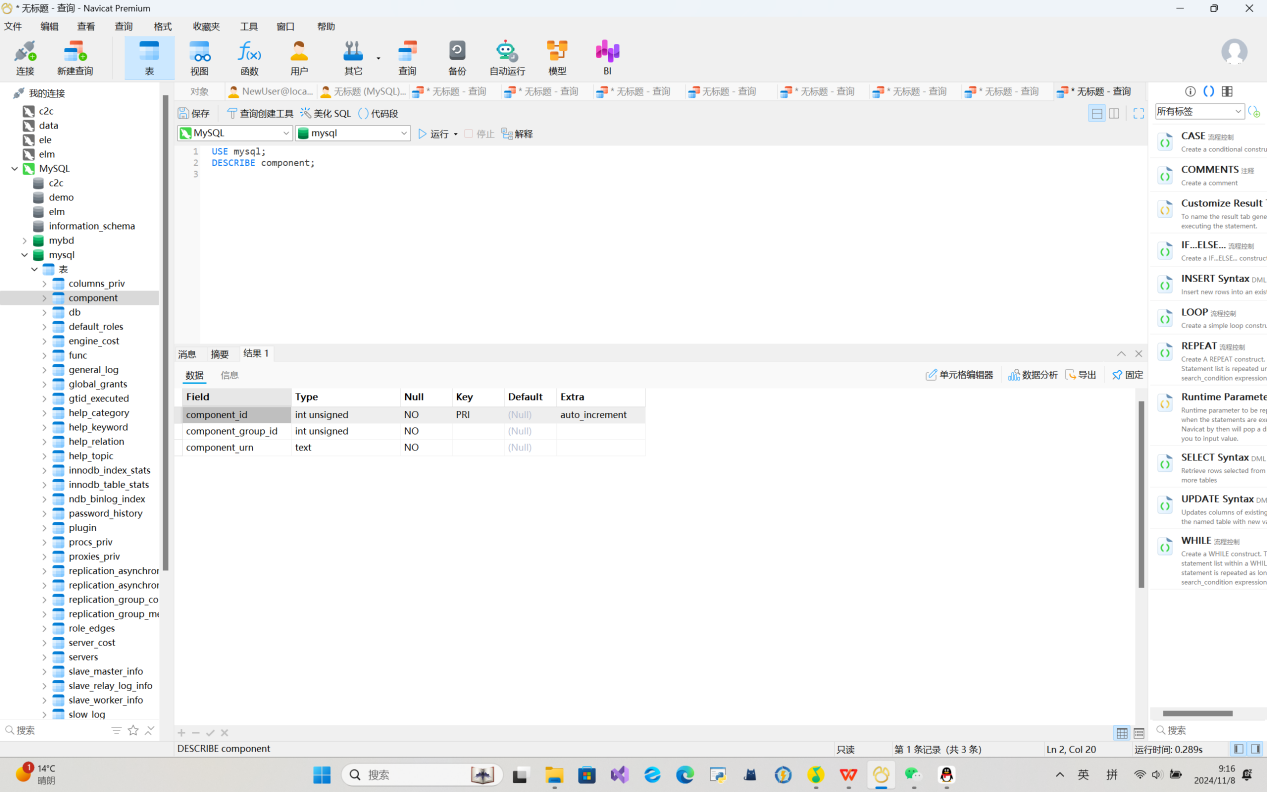
**三、课后练习题**

1、分别用以下几种语句查看数据库“mysql”中任意表的结构或数据（先输入：“USE mysql；”）：

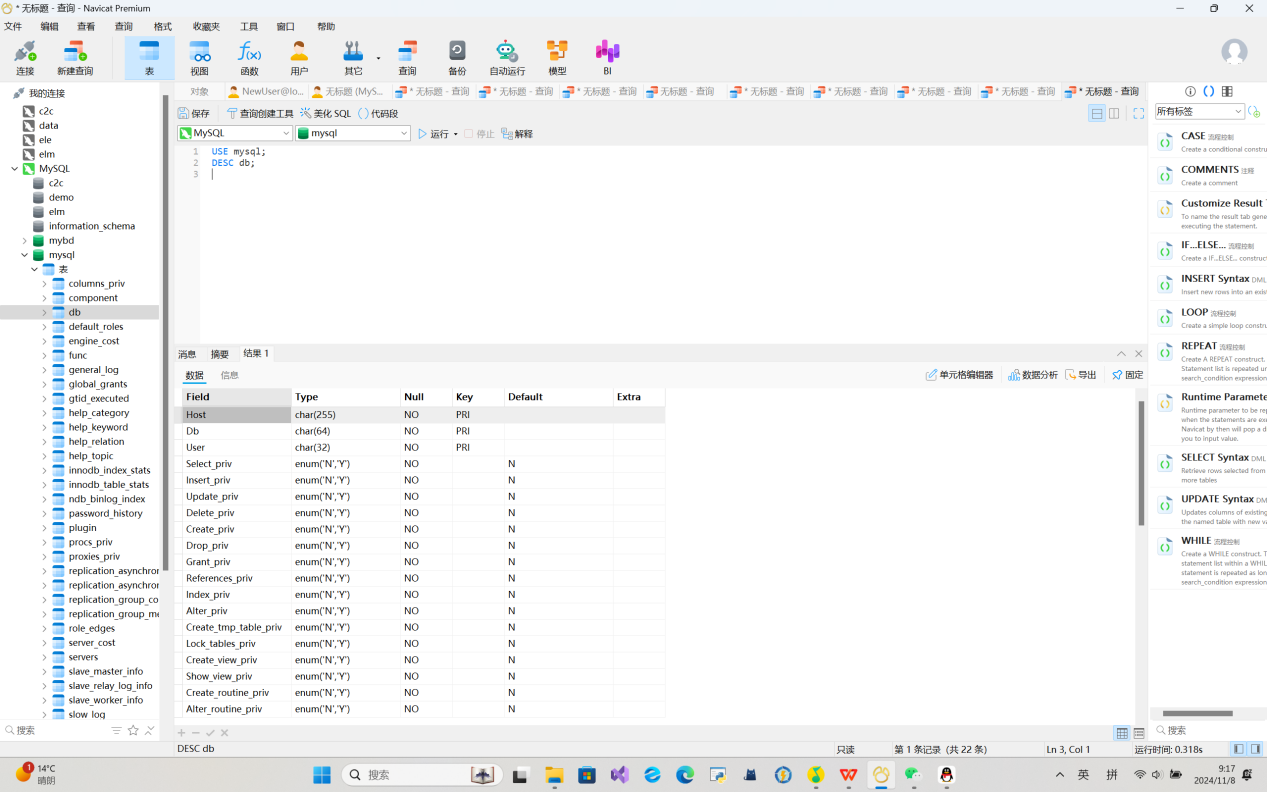
①SHOW COLUMNS FROM 表名;



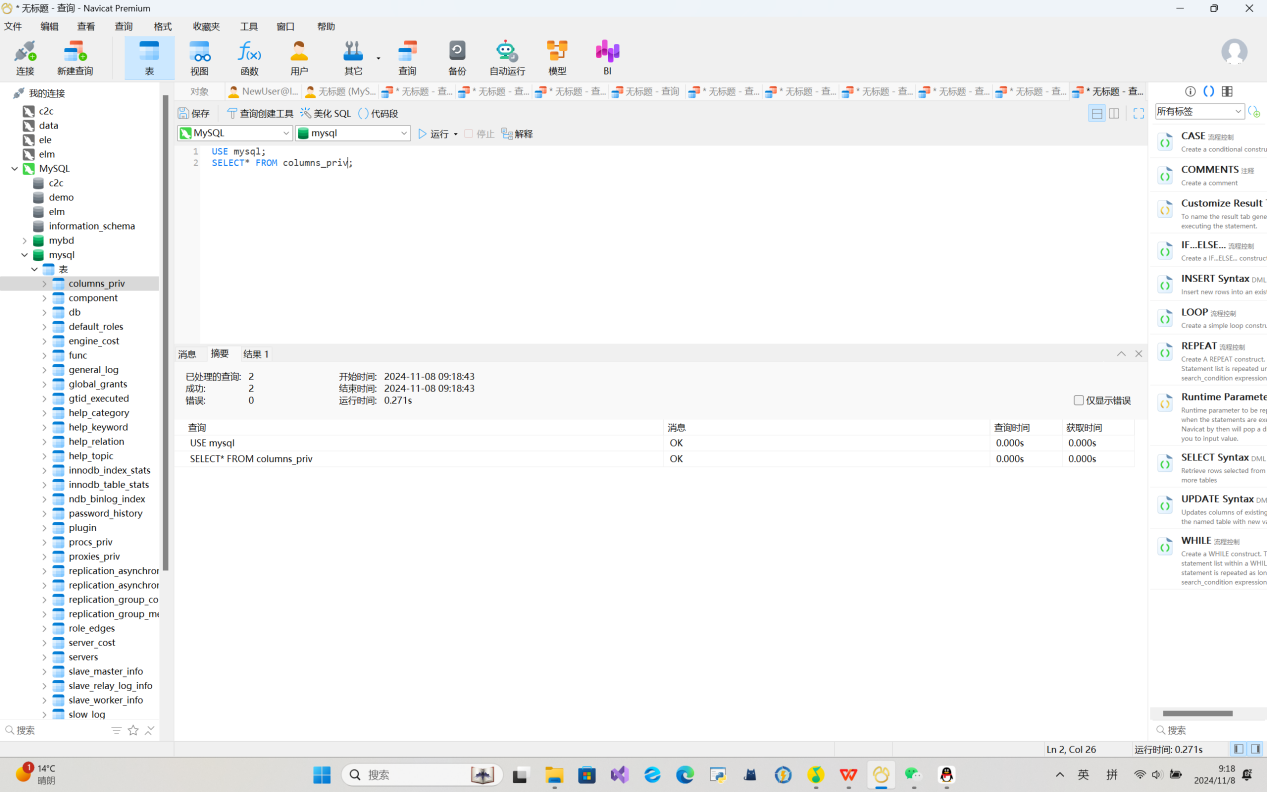
②DESCRIBE 表名;



③DESC 表名



④SELECT \* FROM 表名



2、思考以下问题：

1） mysql中utf8编码的utf8mb3\_bin,utf8mb3\_general\_cs,utf8mb3\_general\_ci三者的区别。

这三个都是 MySQL 中的排序规则（Collation），用于定义如何比较和排序使用 utf8mb3（即原始 utf8 编码）字符集的字符串。它们的主要区别在于对字符的比较方式和区分大小写的处理：

utf8mb3\_bin：二进制排序规则。它直接按二进制值比较字符，这意味着字符完全区分大小写和重音。例如，"a" 和 "A" 是不同的字符。该排序规则比较严格，适合对大小写和重音敏感的数据。

utf8mb3\_general\_cs：这是区分大小写的通用排序规则（general case-sensitive）。它区分大小写，但在某些语言中，它的排序不如特定的语言排序规则准确。例如，"a" 和 "A" 是不同的字符，但可能会忽略重音或某些特定语言的规则。

utf8mb3\_general\_ci：这是不区分大小写的通用排序规则（general case-insensitive）。这种规则不区分大小写，不考虑重音差异。即 "a" 和 "A" 被认为是相同的字符。

总结来说，utf8mb3\_bin 比较严格，区分大小写和重音；utf8mb3\_general\_cs 仅区分大小写，不区分重音；而 utf8mb3\_general\_ci 则既不区分大小写也不区分重音。

2） 不同编码方式的数据库之间进行转储和导入会不会出现错误。

在不同编码方式的数据库之间进行转储和导入可能会出现错误，主要原因包括：

**字符丢失**：如果目标数据库的编码方式无法表示源数据库中的某些字符（例如将 utf8mb4 编码数据导入到 utf8mb3 数据库，utf8mb4 支持的 4 字节字符会丢失），会导致字符缺失或乱码.

**字符集不匹配**：源和目标数据库字符集不匹配时，字符在转储或导入过程中可能被误解或错误转换，导致乱码或数据损坏。

**排序规则冲突**：不同的排序规则会影响字符串比较的结果，尤其是涉及大小写或重音的情况下，可能影响数据的一致性。

因此，在跨编码方式转储和导入数据时，最好确保源和目标数据库的字符集和排序规则一致，或者在转储时指定编码转换参数，确保数据的完整性和正确性。